Mathematics Programs at U.M.B.C.

Mathematics Research Reports

3

Division of Mathematics and Physics

UNIVERSITY OF MARYLAND
BALTIMORE COUNTY

Baltimore, Maryland 21228

18 AFOSR

3

19 TR-79-0060

6

Optimal Estimation for the Satellite Attitude using Star Tracker Measurements

16 2304

17 A1

by

10 James Ting-Ho Lo

14 MRR-

Mathematics Research Report No. 78-14

11 ~~November~~, 1978

12 170 p.

ABSTRACT

An optimal estimation scheme is presented, which determines the satellite attitude using the gyro readings and the star tracker measurements of a commonly used satellite attitude measuring unit. The scheme is mainly based on the exponential Fourier densities that have the desirable closure property under conditioning. By updating a finite and fixed number of parameters, the conditional probability density, which is a exponential Fourier density, is recursively determined.

Simulation results indicate that the scheme is effective and robust. It is believed that this approach is applicable to many other attitude measuring units. As no linearization and approximation are necessary in the approach, it is ideal for systems involving high levels of randomness.

When a system involves little randomness and linearization is not expected to incur much error, the approach can provide a benchmark against which such suboptimal estimators as the extended Kalman filter and the least-squares estimator can be compared. In this spirit, simulated data for HEAO-A were processed to compare the optimal scheme and the extended Kalman filter. The results are presented.

This work was supported by the Goddard Space Flight Center/NASA under Contract No. NAS5-24217 and the Air Force ~~Office of Scientific Research, Air~~ Force Systems Command, USAF, under Grant No. AFOSR-74-2671, NAS5-24217

15

409579

1B

I. Introduction

The project being reported on is mainly concerned with estimating the satellite attitude given the gyro readings and the star tracker measurements of a commonly used satellite attitude measuring unit (SAMU). The SAMU is used in such satellites as the high energy astronomy observatory (HEAO) and the precision pointing control system (PPCS) [1] [2]. It is composed of 3 to 6 rate gyros and 2 star trackers. The satellite attitude is propagated over a certain number of small time intervals by integrating the satellite angular rates determined from the gyro reading. Gyro drift rates, misalignments, and lack of a precise initial attitude reference then make it necessary to employ two gimbaled star trackers to provide a bench mark to the further propagation of the satellite attitude. A star tracker utilizes an image dissector tube to locate the position of a star on its photosensitive surface. Due to the nonstationary nonlinear characteristics of the image dissector deflection coils and the white noise from the processing electronics, it is at this stage that estimation is required.

A new representation of a probability density of a three dimensional rotation called the exponential Fourier density (EFD), was recently introduced [3] [4], which has the desirable closure property under the operation of taking conditional distributions. Using the EFD's, an approach was suggested in [3] [4] to derive recursive formulas for updating the conditional densities of a rotational process given a nonlinear observation in additive white noise.

In this report, this approach is carried out for the aforementioned satellite attitude estimation problem. The recursive formulas for updating the conditional densities of the satellite attitude are derived for arbitrary star tracker equations. These general formulas are included here to accommodate possible future consideration of the distortion characteristics of the image dissector deflection coils [1] [2] and possible future change in the star tracker configuration. These general formulas also provide a basis on which special cases can be easily analyzed. However, they involve a large amount of computation. Their feasibility for on-board implementation is highly questionable.

In a conversation with E. J. Lefferts of the GSFC/NASA, it was observed by him that by choosing appropriately the mathematical description of the star tracker configuration, the star tracker measurement can be expressed in closed form as a linear conbination of the rotational harmonic functions of order one. This observation substantially simplifies the optimal estimation scheme and greatly reduces the amount of computation required in both designing and utilizing the scheme. A detailed derivation of the associated equations is included in the sequel. It is these equations that we use in implementation, simulation, and evaluation.

A total of 57 computer programs were created to carry out essentially the following tasks:

TASK 1. Simulating the satellite attitude propagation and measurement;

TASK 2. Updating the conditional density using the recursive formulas for its Fourier coefficients;

TASK 3. Integrating the conditional covariance matrix of the attitude quaternion;

TASK 4. Computing the maximum eigen-value and its eigenvector of the conditional covariance matrix and the estimation errors;

TASK 5. Searching for the maximum likehood estimate.

TASK 6. Plotting the estimation errors.

A close look at the mathematical models (30) and (31) for the star tracker measurement, which are either used in [1] and [2] or modified by E. J. Lefferts of the GSFC/NASA, revealed that the measurement models are not observable. The nonobservability causes a pseudo-image of each observed star. As the extended K-B filtering is merely a local processing, it does not pick up the pseudo-image and is therefore immune from its effect. In contrast, the optimal scheme does not have any "blind spots" and thus assigns an equal probability to the double images of each of the observed stars. An example illustrating such nonobservability is given in Section VI.

Fortunately enough, this difficulty resulting from the nonobservability can be remedied by introducing a "pseudo-measurement" of the second apparent star direction cosine $u_2(k)$ with respect to the tracker base reference axes. We note that $u_2(k)$ is the component of the direction v :tor $u(k)$ that is perpendicular to the tracker field-of-view and hence can not be measured directly by the tracker. However, from using the satellite attitude estimate $\hat{s}(t)$ at the previous step $t=k-1$, $u_2(k)$ can be predicted, which is to be used as a "measurement" of the real $u_2(k)$. Facilitated with such a pseudo-measurement, the pseudo-image of the observed star can be eliminated. For want of a mathematically rigorous proof, only a heuristic explanation for this pseudo-measurement approach, which is believed to be new, is given in Section VII.

Among the six tasks mentioned above, TASK 3 is most CPU-time-consuming, which involves nine integrations on the 3-dimensional rotation group.

Encouraging is the fast and high concentration of the conditional probability density at the satellite attitude under estimation. This phenomenon is dictated by the theory and suggest two possible ways to get around the difficulty of integration. One way is to localize the integration. Another way is to use the maximum likelihood estimate instead of the optimal estimate. Both methods were researched and implemented on the computer. The simulation results indicate that there is virtually no difference between the estimates obtained in these two ways (at least for the models used in the project).

The maximum likelihood estimator avoids not only integration altogether but also TASK 4-computing the maximum eigenvalue and its eigenvector. All it needs is the updated Fourier coefficients of the conditional density, which are obtained through simple algebraic formulas. Therefore, the maximum likelihood estimator is used in comparison with the extended K-B filter, a standard method for spacecraft attitude estimation.

A comparison between the K-B filtering and our maximum likelihood estimator was conducted by E. J. Lefferts of the GSFC/NASA. A. N. Mansfield of the CSTA generated a sequence of 33 star tracker observations. The average body rates were provided every one third of a second, and the tracker observation was taken every two minutes. The standard deviation of the tracker measurement noises is 20 arcseconds. For such a low noise level, it is known that linearization is a very good approximation. Therefore, it is not surprising that the maximum likelihood estimator is not much better than the K-B filter.

However, the comparison results indicate that the maximum likelihood estimator is almost always better and converges faster than the K-B filter. It is also noted that: (1) The simulated measurement data are in strict accord with the system model, assuming all the true values of the model parameters and the noise statistics are given. (2) There is no random driving term in the state dynamics model for the spacecraft attitude. Under these two conditions in addition to the low measurement noise level mentioned above, even simple-minded estimators can be expected to perform near optimal. But these conditions are usually far from being met in reality.

The simulated examples in Section IX, used to compare the ML and the optimal estimates, werechosen to test the robustness of our new schemes and represent tougher working conditions than the real ones. The simulation results to be depicted in graphs indicate that both the local integration estimator and the maximum likelihood estimator track the signal nicely. Unfortunately, we did not get to use the K-B filter for these cases. But, we believe that the extended K-B filter is no more valid especially for the third example.

The robustness of an estimator toward uncertainty of system parameters and the random driving term in the state dynamics is perhaps the most important consideration in real world application. While there is every reason to believe that the maximum likelihood estimator is superior in this regard, the issue remains to be resolved in hopefully our next project.

A description and a flow chart of each of the computer programs are given in the Appendix. A listing can be obtained from the data set called W9MXN.TESTAL.FORT, which is stored in the IBM360/95 at the GSFC/NASA. The programs involved in each of the aforementioned tasks are listed as follows:

TASK 1.	M05S03 (Main Program), TRANS, NEWO, QTN, QXQ, RANDU
TASK 2.	ADC, COEFF1, DRI2, FU, GAUSS, GAUSS4, NPAQ, INITPQ, FR, SD, C3J, DRI1
TASK 3.	ERR, INTGLE, SINCOS, INSDD, INTGL2, DMNØ, BLKØ, ESTIM
TASK 4.	ERR, EIGEN, EIGVAL, EIGVEC, QVQ, TRIDMX
TASK 5.	MAXI, SEARCH, QTA
TASK 6.	M06PLT, PLOT1, UPDWN

II. Propagation of the Satellite Attitude

The SAMU uses an inertial measuring unit (IMU) composed of 3 to 6 rate gyros, and two star trackers. Let the time t be indexed by a discrete intergration step T. A reading of the IMU is then taken, and compensations are applied by the computer to yield the estimated rate acting upon the satellite. This compensation is based upon the estimates of the gyro misalignments and drift rates. The compensated output, the three estimated angular velocities $[\omega_1,\omega_2,\omega_3]$ of the satellite, is then integrated to yield the estimated satellite attitude quaternion $\rho = [\xi,\eta,\zeta,\chi]^T$ as follows [1] [2]:

$$\rho_{k+1} = Q_k\rho_k \tag{1}$$

where

$$Q = \begin{bmatrix} C_0 & \theta_3 S_0 & -\theta_2 S_0 & \theta_1 S_0 \\ -\theta_3 S_0 & C_0 & \theta_1 S_0 & \theta_2 S_0 \\ \theta_2 S_0 & -\theta_1 S_0 & C_0 & \theta_3 S_0 \\ -\theta_1 S_0 & -\theta_2 S_0 & -\theta_3 S_0 & C_0 \end{bmatrix}$$

$$\theta_0 = T\left(\sum_{i=1}^{3}\omega_i^2\right)^{\frac{1}{2}}$$

$$S_0 = \sin\frac{\theta_0}{2}/\theta_0$$

$$C_0 = \cos\frac{\theta_0}{2}$$

$$\theta_i = \omega_i T, \quad i = 1,2,3.$$

As the IMU parameters can be calibrated very accurately before the satellite is launched and the gyro noise level is extremely low, the estimation error of $(\omega_1,\omega_2,\omega_3)$ is assumed negligible to simplify the project. Hence if the initial attitude were known and if the IMU and computer performed ideally, then the attitude could be predicted precisely for all future time and no additional attitude measuring device would be required. Unfortunately this is not the case and two star trackers are utilized once every N time steps to acquire additional information about the satellite attitude.

As we are only concerned in this project with filtering the star tracker data to estimate the satellite attitude, the attitude propagation between two consecutive star tracker measurements can be combined to yield the following state equation:

$$\underline{\rho}_{k+1} = \underline{Q}_k\underline{\rho}_k \tag{2}$$

where $\underline{\rho}_k = \rho_{kN}$ and $\underline{Q}_k = \prod_{j=0}^{N-1} Q_{kN+j}$

We note that each time step in (2) is NT.

Let $\underline{\rho}_k$ and $\underline{Q}_k$ be identified with their Euler angle representations $s_k(\phi_k,\theta_k,\psi_k)$ and $w_k(\alpha_k,\beta_k,\gamma_k)$ respectively. The equation (2) can thus be rewritten in terms of Euler angles as

$$s_{k+1}(\phi_{k+1},\theta_{k+1},\psi_{k+1}) = w_k(\alpha_k,\beta_k,\gamma_k)\circ s_k(\phi_k,\theta_k,\psi_k) \tag{3}$$

where $\circ$ denotes the product rotation and (α,β,γ) is related to the components $(\underline{Q})_{ij}$ of $\underline{Q}$ through

$$\lambda = [((\underline{Q})_{11}^2 + (\underline{Q})_{21}^2)\ ((\underline{Q})_{41}^2 + (\underline{Q})_{31}^2)]^{\frac{1}{2}} \tag{4}$$

$$\sin\alpha = -\ ((\underline{Q})_{11}(\underline{Q})_{31} + (\underline{Q})_{21}(\underline{Q})_{41})/\lambda$$

$$\cos\alpha = \ ((\underline{Q})_{31}(\underline{Q})_{21} - (\underline{Q})_{11}(\underline{Q})_{41})/\lambda$$

$$\cos\beta = \ 2((\underline{Q})_{11}^2 + (\underline{Q})_{21}^2) - 1, \qquad\qquad 0 \le \beta \le \pi$$

$$\sin\gamma = ((\underline{Q})_{21}(\underline{Q})_{41} - (\underline{Q})_{11}(\underline{Q})_{31})/\lambda$$

$$\cos\gamma = -\ ((\underline{Q})_{11}(\underline{Q})_{41} + (\underline{Q})_{31}(\underline{Q})_{21})/\lambda$$

Hence the time sequence $[\alpha_k,\beta_k,\gamma_k]$ can be easily determined from the satellite angular velocity $[\omega_1,\omega_2,\omega_3]$ through (1) - (4). This sequence will be used in the optimal estimation equations to be derived in the sequel.

III. General Recursive Formulas for Conditional Densities

The star tracker measurements (to be denoted by m_k) are nonlinear functions of the satellite attitude s_k corrupted in additive white noise (to be denoted by v_k). A general mathematical model can be written as

$$m_k = h(s_k,k) + v_k \tag{5}$$

where m_k, $h(s_k k)$, and v_k are all r-vectors. The measurement noise v_k is caused by the processing electronics of the star trackers and it is customarily assumed Gaussian. We write its density function as

$$p(v_k) = (2\pi)^{-\frac{r}{2}}(\det R_k)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\sum_{i,j=1}^{r} R_k^{ij}v_k^i v_k^j\right) \tag{6}$$

where R_k^{ij} is the ixj-th component of the covariance matrix R_k of $v_k = [v_k^1, \ldots v_k^r]^T$.

As suggested in [3] and [4], the function h(·, k) for each k must be approximated very closely by a finite Fourier series. By a slight abuse of the symbol h, it will also be used to denote its Fourier approximate, i.e.

$$h(s_k,k) = \sum_{\ell=0}^{L}\sum_{m,n=-\ell}^{\ell} h_{mn}^{\ell}(k)D_{mn}^{\ell}(s_k) \tag{7}$$

where the rotational harmonic function D_{mn}^{ℓ} is defined, in terms of the Euler angles (ϕ,θ,ψ), by

$$D_{mn}^{\ell}(\phi,\theta,\psi) = \underline{i}^{m-n}\exp[-\underline{i}(m\phi+n\psi)]d_{mn}^{\ell}(\phi) \quad , \underline{i} = (-1)^{\frac{1}{2}} \tag{8}$$

$$d_{mn}^{\ell}(\theta) = \sum_{t}(-1)^{t}\frac{[(\ell+m)!(\ell-m)!(\ell+n)!(\ell-n)!]^{\frac{1}{2}}}{(\ell+m-t)!(t+n-m)!t!(\ell-n-t)!} \cdot\cos^{2\ell+m-n-2t}\frac{\theta}{2}\sin^{2t+n-m}\frac{\theta}{2} \tag{9}$$

As h is a real function, the above Fourier series can be written in the alternative form, called the real form as contrasted to the complex form (7),

$$h(s_k,k) = \frac{1}{2}h_{100}^{\ell}(k)D_{00}^{\ell}(s_k) + \sum_{\substack{m,n=-\ell\\ m<n\\ \text{or } m=n>0}}^{\ell}\left[h_{1mn}^{\ell}(k)\mathrm{Re}D_{mn}^{\ell}(s_k) + h_{2mn}^{\ell}(k)\mathrm{Im}D_{mn}^{\ell}(s_k)\right] \tag{10}$$

where the real coefficients h_{1mn} and h_{2mn} are related with the complex coefficient h^{ℓ}_{mn} through

$$h^{\ell}_{mn} = \frac{1}{2} (h^{\ell}_{1mn} - \underline{i}h^{\ell}_{2mn}) \tag{11}$$

The i-th components of h and h^{ℓ}_{mn} will be denoted by h^{i} and $h^{\ell i}_{mn}$ respectively.

A set of Fortran IV programs to compute h^{ℓ}_{mn}'s and the approximation error was written and successfully tested on the IBM360/95 at the GSFC/NASA. Descriptions and flow charts of the programs are given in the Appendix.

It is assumed that the conditional density of s_{k-1} given $m^{k-1}: = \{m_1,\ldots,m_{k-1}\}$ is an exponential Fourier density of order 2L, i.e.

$$p(s_{k-1}|m^{k-1}) = \exp \sum_{\ell=0}^{2L} \sum_{m,n=-\ell}^{\ell} p^{\ell,k-1}_{mn} \; D^{\ell}_{mn}(s_{k-1}) \tag{12}$$

for k equal to a positive integer. We note that a rotational normal density is an exponential Fourier density of order 1. The reason for choosing the order 2L will become clear in the following derivation of the recursive formulas for $p^{\ell k}_{mn}$.

By Bayes' rule, we have

$$p(s_k|m^k) = c_k p(m_k|s_k)p(s_k|m^{k-1}) \tag{13}$$

c_k = a normalizing constant

As w_k is a group element, from (3) it follows that $s_{k-1} = w_k^{-1} \circ s_k$

Substituting this into (12) yields

$$\begin{aligned} p(s_k|m^{k-1}) &= \exp \sum p_{mn}^{\ell,k-1} D_{mn}^{\ell}(w_{k-1}^{-1} \circ s_k) \\ &= \exp \sum p_{mn}^{\ell,k-1} \sum_{q=-1}^{\ell} D_{mq}^{\ell}(w_{k-1}^{-1}) D_{qn}^{\ell}(s_k) \\ &= \exp \sum \Big(\sum_{q=-\ell}^{\ell} p_{qn}^{\ell,k-1} D_{qm}^{\ell}(w_{k-1}^{-1}) \Big) D_{mn}^{\ell}(s_k) \end{aligned} \tag{14}$$

where $\sum = \sum_{\ell=0}^{2L} \sum_{m,m=-\ell}^{\ell}$ and the second equality holds because D_{mn}^{ℓ} is a matrix representation of the three dimensional rotation group.

From (5), (6), and the definition of a conditional density, it is clear that

$$\begin{aligned} p(m_k|s_k) &= (2\pi)^{-\frac{r}{2}} (\det R_k)^{-\frac{1}{2}} \exp\Big[-\frac{1}{2} \sum_{i,j=1}^{r} R_k^{ij} (m_k^i - h^i(s_k,k))(m_k^j - h^j(s_k,k)) \Big] \\ &= c_k' \exp\Big[-\frac{1}{2} \sum_{i,k=1}^{r} R_k^{ij} \Big(m_k^i - \sum_{\ell=0}^{L} \sum_{m,n=-\ell}^{\ell} h_{mn}^{\ell i}(k) D_{mn}^{\ell}(s_k)\Big)\Big(m_k^j - \sum_{\ell'=0}^{L} \sum_{m'n'=-\ell'}^{\ell'} h_{m'n'}^{\ell}(k) D_{m'n'}^{\ell}(s_k)\Big) \Big] \\ &= c_k' \exp\Big[C_0 + \sum_{\ell=0}^{L} \sum_{m,n=-\ell}^{\ell} C_{mn}^{\ell} D_{mn}^{\ell}(s_k) + \\ &\qquad \sum_{\ell,\ell'=0}^{L} \sum_{m,n=-\ell}^{\ell} \sum_{m',n'=-\ell'}^{\ell'} C^{\ell\ell'}(m,n,m',n') D_{mn}^{\ell}(s_k) D_{m'n'}^{\ell'}(s_k) \Big] \end{aligned} \tag{15}$$

where

$$C_0 = -\frac{1}{2} \sum_{i,j=1}^{r} R_k^{ij}\, m_k^i\, m_k^j$$

$$C_{mn}^{\ell} = \frac{1}{2} \sum_{i,j=1}^{r} R_k^{ij} (m_k^i h_{mn}^{\ell j}(k) + m_k^j h_{mn}^{\ell i}(k)) = \sum_{i,j=1}^{r} R_k^{ij} m_k^i h_{mn}^{\ell j}(k) \tag{16}$$

$$C^{\ell\ell'}(m,n,m',n') = -\frac{1}{2}\sum_{i,j=1}^{r} R_k^{ij} h_{mn}^{\ell i}(k) h_{m'n'}^{\ell' j}(k) \qquad (17)$$

Furthermore, it is known [5, p.160] that

$$D_{mn}^{\ell}(s_k) D_{m'n'}^{\ell'}(s_k) = \sum_{q=q_0}^{\ell+\ell'} (2q+1)(-1)^{m''-n''} \begin{pmatrix} \ell & \ell' & q \\ m & m' & -m'' \end{pmatrix} \begin{pmatrix} \ell & \ell' & q \\ n & n' & -n'' \end{pmatrix} D_{m''n''}^{q}(s_k) \qquad (18)$$

$$m'' = m + m'$$

$$n'' = n + n'$$

$$q_0 = \max\left(|\ell-\ell'|, |m''|, |n''|\right)$$

$$\begin{pmatrix} j & k & \ell \\ m & n & p \end{pmatrix} = (-1)^{2j-k+n} \left[\frac{(j+k-\ell)!(k+\ell-j)!(\ell+j-k)!(\ell+p)!(\ell-p)!}{(j+k+\ell+1)!(j+m)!(j-m)!(k+n)!(k-n)!} \right]^{\frac{1}{2}}$$

$$\cdot \sum_t (-1)^t \frac{(\ell+j-n-t)!(k+n+t)!}{(\ell+p-t)!(t+k-j-p)!t!(\ell-k+j-t)!} \qquad (19)$$

where the sum on t is over values such that the arguments of all the factorial functions are nonnegative. Let us define the symbols:

$$\gamma^q \begin{pmatrix} \ell & m & n \\ \ell' & m' & n' \end{pmatrix} := C^{\ell\ell'}(m,n,m',n')(2q+1)(-1)^{m''-n''} \begin{pmatrix} \ell & \ell' & q \\ m & m & -m'' \end{pmatrix} \begin{pmatrix} \ell & \ell' & q \\ n & n' & -n'' \end{pmatrix} \qquad (20)$$

$$\Gamma^q(\ell,\ell',m'',n'') := \sum_{\substack{-\ell\le m,n\le\ell \\ m''-\ell\le m\le m''+\ell' \\ n''-\ell\le n\le n''+\ell'}} \gamma^q \begin{pmatrix} \ell & m & n \\ \ell' & m'' & n''-n \end{pmatrix} \qquad (21)$$

$$\Gamma^q_{mn} \; : = \sum_{\substack{\ell,\ell'=0 \\ |\ell-\ell'| \leq q \leq \ell+\ell'}}^{L} \Gamma^q(\ell,\ell',m,n) \tag{22}$$

Using essentially (18), the last term in (15) is transformed into a linear combination of the rotational harmonics as follows:

$$\sum_{\ell,\ell'=0}^{L} \sum_{m,n=-\ell}^{\ell} \sum_{m',n'=-\ell'}^{\ell'} C^{\ell\ell'}(m,n,m',n') D^{\ell}_{mn}(s_k) D^{\ell'}_{m'n'}(s_k)$$

$$\sum_{\ell,\ell'=0}^{L} \sum_{m,n=-\ell}^{\ell} \sum_{m',n'=-\ell'}^{\ell'} \sum_{q=q_0}^{\ell+\ell'} \gamma^q \begin{pmatrix} \ell & m & n \\ \ell' & m' & n' \end{pmatrix} D^q_{m''n''}(s_k)$$

$$\sum_{q=0}^{2L} \sum_{m'',n''=-q}^{q} \left[\sum_{\substack{\ell,\ell'=0 \\ |\ell-\ell'| \leq q \leq \ell+\ell'}}^{L} \Gamma^q(\ell,\ell',m'',n'') \right] D^q_{m''n''}(s_k)$$

$$\sum_{q=0}^{2L} \sum_{m,n=-q}^{q} \Gamma^q_{mn} D^q_{mn}(s_k). \tag{23}$$

Substituting (22), (15), and (14) into (13), we obtain

$$p(s_k|m^k) = \exp\left[\sum_{\ell=0}^{2L} \sum_{m,n=-\ell}^{\ell} \left(\sum_{q=-\ell}^{\ell} p^{\ell,k-1}_{qn} D^{\ell}_{qn}(w^{-1}_{k-1}) + \Gamma^{\ell}_{mn} + \chi_{[0,L]}(\ell)\, C^{\ell}_{mn} \right) D^{\ell}_{mn}(s_k) \right] , \tag{24}$$

the constants c_k, c'_k, and C_0 being absorbed into the coefficient of $D^0_{00}(s_k) = 1$, which is a normalizing constant, and $\chi_{[0,L]}$ is the characteristic function for $[0,L]$.

We note that $p(s_k|m^k)$ is also an exponential Fourier density of order 2L. Comparing (12) and (24), we obtain the recursive formula for $p^{\ell k}_{mn}$:

$$p^{\ell k}_{mn} = \sum_{q=-\ell}^{\ell} p^{\ell,k-1}_{qn} D_{qm}(w^{-1}_{k-1}) + \Gamma^{\ell}_{mn} + \chi_{[0,L]}(\ell) C^{\ell}_{mn} \tag{25}$$

where Γ and C are defined by (20)-(22) and (16).

The function enclosed in the brackets in (24) is a real function. It is often easier to express it in the real form of the Fourier series. Using the notational convention established in (10) and (11), the equation (24) can be written as

$$p(s_k|m^k) = \exp \sum_{\ell=0}^{2L} \left[\frac{1}{2} p^{\ell k}_{100} D^{\ell}_{00}(s_k) + \sum_{\substack{m,n=-\ell \\ m<n \\ \text{or } m=n>0}}^{\ell} \left(p^{\ell k}_{1mn} \mathrm{Re} D^{\ell}_{mn}(s_k) + p^{\ell k}_{2mn} \mathrm{Im} D^{\ell}_{mn}(s_k) \right) \right] \tag{26}$$

$\frac{1}{2} p^{0k}_{100}$ = the normalizing constant

The recursive formulas for p^{k}_{1mn} and p^{k}_{2mn} can be deduced from (25) as follows:

$$\sum_{q=-\ell}^{\ell} p^{\ell,k-1}_{qn} D^{\ell}_{qn}(w^{-1}_{k-1}) = \sum_{q=-\ell}^{\ell} \frac{1}{2} \left(p^{\ell,k-1}_{1qn} - \underline{i} p^{\ell,k-1}_{2qn} \right) \left(\mathrm{Re} D^{\ell}_{qm}(w^{-1}_{k-1}) + \underline{i} \mathrm{Im} D^{\ell}_{qm}(w^{-1}_{k-1}) \right)$$

$$= \sum_{q=-\ell}^{\ell} \frac{1}{2} \left[\left(p_{1qn}^{\ell,k-1} \mathrm{Re} D_{qm}^{\ell}(w_{k-1}^{-1}) + p_{2qn}^{\ell,k-1} \mathrm{Im} D_{qm}^{\ell}(w_{k-1}^{-1}) \right) \right.$$

$$\left. + \underline{i} \left(p_{1qn}^{\ell,k-1} \mathrm{Im} D_{qm}^{\ell}(w_{k-1}^{-1}) - p_{2qn}^{\ell,k-1} \mathrm{Re} D_{qm}^{\ell}(w_{k-1}^{-1}) \right) \right]$$

$$C_{mn}^{\ell} = \frac{1}{2} \sum_{i,j=1}^{r} (R_k^{ij} m_k^i h_{1mn}^{\ell j} - \underline{i}\, R_k^{ij} m_k^i h_{2mn}^{\ell j})$$

$$\Gamma_{mn}^{\ell} = \sum_{\substack{q,q'=0 \\ |q-q'| \le \ell \le q+q'}}^{L} \Gamma^{\ell}(q,q',m,n) = \sum_{\substack{q,q'=0 \\ |q-q'| \le \ell \le q+q'}}^{L} \sum_{\substack{-q \le t,s \le q \\ m-q' \le t \le m+q' \\ n-q' \le s \le n+q'}} \gamma^{\ell}\begin{pmatrix} q & t & s \\ q' & m-t & n-s \end{pmatrix}$$

$$\gamma^{\ell}\begin{pmatrix} q & t & s \\ q' & m-t & n-s \end{pmatrix} = (-1)^{m-n}(2\ell+1) \begin{pmatrix} q & q' & \ell \\ t & m-t & -m \end{pmatrix} \begin{pmatrix} q & q' & \ell \\ s & n-s & -n \end{pmatrix} C^{qq'}(t,s,m-t,n-s)$$

$$C^{qq'}(t,s,m-t,n-s) = -\frac{1}{8} \sum_{i,j=1}^{r} R_k^{ij} \left[\left(h_{1ts}^{qi} h_{1,m-t,n-s}^{q'j} - h_{2ts}^{qi} h_{2,m-t,n-s}^{q'j} \right) \right.$$

$$\left. - \underline{i} \left(h_{2ts}^{qi} h_{1,m-t,n-s}^{q'j} + h_{1ts}^{qi} h_{2,m-t,n-s}^{q'j} \right) \right]$$

$$p_{1mn}^{\ell k} = 2\mathrm{Re}\, p_{mn}^{\ell k} \quad , \quad p_{2mn}^{\ell k} = -2\mathrm{Im}\, p_{mn}^{\ell k}$$

$$p_{1mn}^{\ell k} = \sum_{q=-\ell}^{\ell} \left(p_{1qn}^{\ell,k-1} \mathrm{Re} D_{qm}^{\ell}(w_{k-1}^{-1}) + p_{2qn}^{\ell,k-1} \mathrm{Im} D_{qm}^{\ell}(w_{k-1}^{-1}) \right)$$

$$+ \chi_{[0,L]}(\ell) \sum_{i,j=1}^{r} R_k^{ij} m_k^i h_{1mn}^{\ell j} + \sum_{\substack{q,q'=0 \\ |q-q'| \le \ell \le q+q'}}^{L} \sum_{\substack{-q \le t,s \le q \\ m-q' \le t \le q'+m \\ n=q' \le s \le n+q'}} \left[(-1)^{m-n+1} \frac{2\ell+1}{4} \right.$$

$$\cdot \begin{pmatrix} q & q' & \ell \\ t & m-t & -m \end{pmatrix} \begin{pmatrix} q & q' & \ell \\ s & n-s & -n \end{pmatrix} \sum_{i,j=1}^{r} R_k^{ij} \left(h_{1ts}^{qi} h_{1,m-t,n-s}^{q'j} - h_{2ts}^{qi} h_{2,m-t,n-s}^{q'j} \right) \Bigg] \quad (27)$$

$$p_{2mn}^{\ell k} = \sum_{q=-\ell}^{\ell} \left(p_{2qn}^{\ell,k-1} \operatorname{Re} D_{qm}^{\ell}(w_{k-1}^{-1}) - p_{1qn}^{\ell,k-1} \operatorname{Im} D_{qm}^{\ell}(w_{k-1}^{-1}) \right)$$

$$+ \chi_{[0,L]}(\ell) \sum_{i,j=1}^{r} R_k^{ij} m_k^i h_{2mn}^{\ell j} + \sum_{\substack{q,q'=0 \\ |q-q'| \le \ell \le q+q'}}^{L} \sum_{\substack{-q \le t,s \le q \\ m-q' \le t \le q'+m \\ n-q' \le s \le q'+n}} \Bigg[(-1)^{m-n+1} \frac{2\ell+1}{4}$$

$$\cdot \begin{pmatrix} q & q' & \ell \\ t & m-t & -m \end{pmatrix} \begin{pmatrix} q & q' & \ell \\ s & n-s & -n \end{pmatrix} \sum_{i,j=1}^{r} R_k^{ij} \left(h_{2ts}^{qi} h_{1,m-t,n-s}^{q'j} + h_{1ts}^{qi} h_{2,m-t,n-s}^{q'j} \right) \Bigg] \quad (28)$$

IV. Star Tracker Equations in Terms of Harmonics

A star tracker is a telescope-detector system mounted with two degrees of freedom to the satellite. The field of view of the detector is a 8 by 8 degree window whose center point is on the optical axis and whose plane is normal to this axis. A star appearing within this window is sensed by the detector and, after some electronic processing, results in the output of two voltages representing the position of the star within the window along two mutually perpendicular axes.

When a star tracker observation is taken at time k, the star is identified from a star map and its apparent direction cosines $a(k) = [a_1(k), a_2(k), a_3(k)]^T$ with respect to the earth-centered inertial set of coordinate system can be easily computed from the absolute velocity of the satellite. The apparent star direction cosines $u(k) = [u_1(k), u_2(k), u_3(k)]^T$ with respect to the tracker base reference axes are related to a as follows:

$$u(k) = A_{x^t x} A_{xz^I}(k) a(k) \tag{29}$$

where $A_{x^t x}$ is the constant 3 x 3 orthogonal matrix representing the orientation of the tracker base reference axes with respect to the satellite base reference axes and $A_{xz^I}(k)$ is the 3 x 3 orthogonal matrix representing the satellite attitude with respect to the earth-centered inertial set at time k.

The mathematical model used [1] and [2] for the relationship between the tracker output voltages $[y_1(1), y_3(k)]$ and u(k) is the following:

$$\begin{bmatrix} y_1(k) \\ y_3(k) \end{bmatrix} = \begin{bmatrix} -u_1(k)/u_2(k) \\ u_3(k)/u_2(k) \end{bmatrix} + \begin{bmatrix} n_1(k) \\ n_3(k) \end{bmatrix} \tag{30}$$

With $[n_1(k), n_3(k)]$ denoting two independent white Gaussian noise, the model is only an approximation. It was pointed out by E. J. Lefferts of the GSFC/NASA that the following model provides just as good an approximation to the relationship between $[y_1(k), y_3(k)]$ and u(k):

$$\begin{bmatrix} v_1(k) \\ y_3(k) \end{bmatrix} = \begin{bmatrix} -u_1(k) \\ u_3(k) \end{bmatrix} + \begin{bmatrix} n_1(k) \\ n_3(k) \end{bmatrix} \tag{31}$$

It will be seen later that the components of A_{xz^I} are simple linear combinations of the rotational hermonics of order 1. Whence so are the components of u(k). Using the model (31), we do not need to use the set of Fortran IV programs provided in the Appendix to compute $h^{\ell}_{mn}(k)$'s in (7) to approximate $h(s_k,k)$. Above all, the star observed by the star tracker changes from time to time and thus $h(s_k,k)$ is a time-varing function. Using the Fortran programs to compute $h_{mn}(k)$'s for each star observation may amount to unbearable computational burden, especially when the estimation has to be done on board the satellite. The use of the model (31) not only eliminates such a difficulty but also keeps minimal the number of harmonics required to update the conditional densities of the satellite attitude. In the following, we will restrict our attention to the model (31).

The SAMU uses two star trackers. We will use a single underline and a double underline to refer to Star Tracker I and Star Tracker II, respectively, e.g., $(\underline{\underline{u}}_1, \underline{\underline{u}}_2, \underline{\underline{u}}_3)$ denotes the apparent star direction cosines in the Star Tracker II base reference axes $(\bar{x}_1^2, \bar{x}_2^2, \bar{x}_3^2)$. When the underlines are omitted, the equations are valid for both star trackers.

We pool the measurements from the two star trackers and write

$$\begin{bmatrix} m_k^1 \\ m_k^2 \\ m_k^3 \\ m_k^4 \end{bmatrix} = m_k = h(s_k,k) + v_k = \begin{bmatrix} h^1(s_k,k) \\ h^2(s_k,k) \\ h^3(s_k,k) \\ h^4(s_k,k) \end{bmatrix} + \begin{bmatrix} v_k^1 \\ v_k^2 \\ v_k^3 \\ v_k^4 \end{bmatrix} . \tag{32}$$

where

$$\begin{bmatrix} h^1(s_k,k) \\ h^2(s_k,k) \\ h^3(s_k,k) \\ h^4(s_k,k) \end{bmatrix} := \begin{bmatrix} -\underline{u}_1(s_k,k) \\ \underline{u}_3(s_k,k) \\ -\underline{\underline{u}}_1(s_k,k) \\ \underline{\underline{u}}_3(s_k,k) \end{bmatrix} \tag{33}$$

We will now express u_j as a linear combination of the rotational harmonics of order 1: We recall that

$$d^{\ell}_{mn}(\theta) = \sum_{t=\max(0,m-n)}^{\min(\ell-n,\ell+m)} (-1)^t \frac{[(\ell+m)!(\ell-m)!(\ell+n)!(\ell-n)!]^{\frac{1}{2}}}{(\ell+m-t)!(t+n-m)!t!(\ell-n-t)!}$$

$$\cdot \cos^{2\ell+m-n-2t}\frac{\theta}{2}\,\sin^{2t+n-m}\frac{\theta}{2} \qquad (34)$$

$$D^{\ell}_{mn}(\phi,\theta,\psi) = \exp\left[\left(\frac{\pi}{2}-\phi\right)m - \left(\frac{\pi}{2}+\psi\right)n\right] d^{\ell}_{mn}(\theta) \qquad (35)$$

whence

$$\begin{bmatrix} \mathrm{Im}D^1_{11} & \mathrm{Re}D^1_{-10} & \mathrm{Re}D^1_{-11} \\ \mathrm{Im}D^1_{-10} & D^1_{00} & \mathrm{Re}D^1_{01} \\ \mathrm{Im}D^1_{-11} & \mathrm{Im}D^1_{01} & \mathrm{Re}D^1_{11} \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}\sin(\phi+\psi)(1+\cos\theta) & \frac{1}{\sqrt{2}}\sin\phi\sin\theta & \frac{1}{2}\cos(\phi-\psi)(1-\cos\theta) \\ -\frac{1}{\sqrt{2}}\cos\phi\sin\theta & \cos\theta & -\frac{1}{\sqrt{2}}\sin\psi\sin\theta \\ -\frac{1}{2}\sin(\phi-\psi)(1-\cos\theta) & -\frac{1}{\sqrt{2}}\cos\psi\sin\theta & \frac{1}{2}\cos(\phi+\psi)(1+\cos\theta) \end{bmatrix}$$

The matrix A_{xz^I} can then be expressed in terms of the harmonics as follows:

$$A_{xz^I} = \begin{bmatrix} \cos\phi\cos\psi-\sin\phi\cos\theta\sin\psi & \cos\psi\sin\phi+\sin\psi\cos\phi\cos\theta & \sin\psi\sin\theta \\ -\sin\psi\cos\phi-\cos\psi\sin\phi\cos\theta & -\sin\phi\sin\psi+\cos\phi\cos\psi\cos\theta & \cos\psi\sin\theta \\ \sin\phi\sin\theta & -\cos\phi\sin\theta & \cos\theta \end{bmatrix}$$

$$= \begin{bmatrix} \mathrm{Re}D^1_{11}-\mathrm{Re}D^1_{-11} & -\mathrm{Im}D^1_{-11}-\mathrm{Im}D^1_{11} & -\sqrt{2}\mathrm{Re}D^1_{01} \\ -\mathrm{Im}D^1_{-11}+\mathrm{Im}D^1_{11} & \mathrm{Re}D^1_{11}+\mathrm{Re}D^1_{-11} & -\sqrt{2}\mathrm{Im}D^1_{01} \\ \sqrt{2}\mathrm{Re}D^1_{-10} & \sqrt{2}\mathrm{Im}D^1_{-10} & D^1_{00} \end{bmatrix} \qquad (36)$$

Let the ixj-th component of the constant matrix $A_{x^t x}$ be denoted by m_{ij}. From (29), it follows that, for j = 1,2,3 and for both star trackers,

$$\begin{aligned} u_j(k) = {} & m_{j3}a_3(k)D^1_{00}(s_k) + \Big[\sqrt{2}m_{j3}a_1(k)\mathrm{Re}D^1_{-10}(s_k) + (m_{j2}a_2(k) - m_{j1}a_1(k)\mathrm{Re}D^1_{-11}(s_k) \\ & - \sqrt{2}m_{j1}a_3(k)\mathrm{Re}D^1_{01}(s_k) + (m_{j1}a_1(k) + m_{j2}a_2(k))\mathrm{Re}D^1_{11}(s_k) \Big] \\ & + \Big[\sqrt{2}m_{j3}a_2(k)\mathrm{Im}D^1_{-1}(s_k) - (m_{j1}a_2(k) + m_{j2}a_1(k)\mathrm{Im}D^1_{-11}(s_k) \\ & - \sqrt{2}m_{j2}a_3(k)\mathrm{Im}D^1_{01}(s_k) + (m_{j2}a_1(k) - m_{j1}a_2(k))\mathrm{Im}D^1_{11}(s_k) \Big] \end{aligned} \tag{37}$$

The Fourier coefficients, $h^{\ell i}_{1mn}(k)$'s and $h^{\ell i}_{2mn}(k)$'s for $h^i(s_k,k)$ can be easily identified from (33) and (37) and will not be displayed here. We only note that they can be very easily determined from the star direction a(k) and the recursive formulas are then readily applicable.

V. Updating the Conditional Densities Using Star Tracker Measurements

We will now display the recursive formulas for the conditional densities of the satellite to be used in the next two modules of this project. The star tracker equations are (32), (33), and (37) developed in the previous section. We assume that the measurement noise process v_k has statistically independent components, i.e., $R^{ij}_k=0$, if $i \neq j$.

We recall that the function h^i in (33) is a linear combination of the first order harmonics, $\mathrm{Re}D^1_{mn}$ and $\mathrm{Im}D^1_{mn}$. Hence $h^{qi}_{1mn}=h^{qi}_{2mn}=0$ unless $q=1$. This observation greatly simplifies the multiple summation sign in front of the brackets in (27) and (28):

$$\sum_{\substack{q,q'=0 \\ |q-q'|\leq\ell\leq q+q'}} \quad \sum_{\substack{-q\leq t,s\leq q \\ m-q'\leq t\leq m+q' \\ n-q'\leq s\leq n+q'}} = \sum_{\substack{\max(-1,m-1)\leq t\leq\min(1,m+1) \\ \max(-1,n-1)\leq s\leq\min(1,n+1)}}$$

In view of the assumption that $R_k^{ij}=0$, for $i\neq j$, we obtain from (27) and (28) the following recursive formulas: for $\ell=1$, 2 and $m,n = -\ell, \ldots, \ell$,

$$\begin{aligned} p_{1mn}^{\ell k} = {} & \sum_{q=-\ell}^{\ell} p_{1qn}^{\ell,k-1}\mathrm{Re}D_{qm}^{\ell}(w_{k-1}^{-1}) + p_{2qn}^{\ell,k-1}\mathrm{Im}D_{qm}^{\ell}(w_{k-1}^{-1}) \\ & + \chi_{[0,L]}(\ell)\sum_{j=1}^{4} R_k^{jj}m_k^j h_{1mn}^{\ell j} + \sum_{\substack{\max(-1,m-1)\leq t\leq\min(1,m+1) \\ \max(-1,n-1)\leq s\leq\min(1,n+1)}} \Bigg[(-1)^{m-n+1} \\ & \cdot \left(\frac{2\ell+1}{4}\right) \begin{pmatrix} 1 & 1 & \ell \\ t & m-t & -m \end{pmatrix} \begin{pmatrix} 1 & 1 & \ell \\ s & n-s & -n \end{pmatrix} \sum_{j=1}^{4} R_k^{jj} \Big(h_{1ts}^{1j} h_{1,m-t,n-s}^{1j} \\ & - h_{2ts}^{1j} h_{2,m-t,n-s}^{1j} \Big) \Bigg] \end{aligned} \tag{38}$$

$$\begin{aligned} p_{2mn}^{\ell k} = {} & \sum_{q=-\ell}^{\ell} \left(p_{2qm}^{\ell,k-1}\mathrm{Re}D_{qm}^{\ell}(w_{k-1}^{-1}) - p_{1qn}^{\ell,k-1}\mathrm{Im}D_{qm}^{\ell}(w_{k-1}^{-1}) \right) \\ & + \chi_{[0,L]}(\ell) \sum_{j=1}^{4} R_k^{jj}m_k^j h_{2mn}^{\ell j} + \sum_{\substack{\max(-1,m-1)\leq t\leq\min(1,m+1) \\ \max(-1,n-1)\leq s\leq\min(1,n+1)}} \Bigg[(-1)^{m-n+1} \\ & \cdot \left(\frac{2\ell+1}{4}\right) \begin{pmatrix} 1 & 1 & \ell \\ t & m-t & -m \end{pmatrix} \begin{pmatrix} 1 & 1 & \ell \\ s & n-s & -n \end{pmatrix} \sum_{j=1}^{4} R_k^{jj} \Big(h_{2ts}^{1j} h_{1,m-t,n-s}^{1j} \\ & + h_{1ts}^{1j} h_{2,m-t,n-s}^{1j} \Big) \Bigg] \end{aligned} \tag{39}$$

where, from (19),

$$\begin{pmatrix} 1 & 1 & \ell \\ t & m-t & -m \end{pmatrix} = (-1)^{1+m-t} \left[\frac{\ell!\ell!(\ell-m)!(\ell+m)!}{(3+\ell)!(1+t)!(1-t)!(1+m-t)!(1-m+t)!} \right]^{\frac{1}{2}}$$

$$\cdot \sum_{r=\max(0,-m)}^{\min(\ell-m,\ell)} (-1)^r \frac{(\ell+1-m+t-r)!(1+m-t+r)!}{(\ell-m-r)!(r+m)!r!(\ell-r)!} \qquad (40)$$

The conditional density $p(s_k/m^k)$ is then obtained through (26):

$$p(s_k|m^k) = \exp \sum_{\ell=0}^{2} \left[\frac{1}{2} p^{\ell k}_{100} D^{\ell}_{00}(s_k) + \sum_{\substack{m,n=-\ell \\ m<n \\ \text{or } m=n>0}}^{\ell} \left\{ p^{\ell k}_{1mn} \mathrm{Re} D^{\ell}_{mn}(s_k) + p^{\ell k}_{2mn} \mathrm{Im} D^{\ell}_{mn}(s_k) \right\} \right] \qquad (26)$$

$\frac{1}{2} p^{0k}_{100}$ = the normalizing constant

It is noted that there are 35 Fourier coefficients, $p^{\ell k}_{1mn}$ and $p^{\ell k}_{2mn}$, which are recursively computed by reasonably simple scalar equations (38) and (39).

II. Nonobservability of the Star Tracker Equations

We consider the mathematical model (31), to be repeated in the following, for the star tracker measurement in this section. Our argument here applies also to the model (30). We recall that the model (31) is

$$\begin{bmatrix} y_1(k) \\ y_3(k) \end{bmatrix} = \begin{bmatrix} -u_1(s_k,k) \\ u_3(s_k,k) \end{bmatrix} + \begin{bmatrix} n_1(k) \\ n_3(k) \end{bmatrix} \qquad (31)$$

$$u(s_k,k) = A_{x^t x} A_{xz^I}(k)a(k) \qquad (29)$$

where a(k) is the apparent direction cosines of the observed star with respect to the earth-centered inertial set z^I of coordinate system, $A_{x^t x}$ is the constant 3x3 orthogonal matrix representing the orientation of the tracker base reference axes x^t with respect to the satellite base reference axes x, and $A_{xz^I}(k)$ is the 3x3 orthogonal matrix representing the satellite attitude with respect to the earth-centered inertial set at time k.

The SAMU uses two star trackers. We will use a single underline and a double underline to refer to Star Tracker I and Star Tracker II, respectively. When the underlines are omitted, the equations are valid for both star trackers.

We pool the measurements from the two star trackers and write, suppressing the time variable k,

$$\begin{bmatrix} m^1 \\ m^2 \\ m^3 \\ m^4 \end{bmatrix} = m = h(s) + v = \begin{bmatrix} h^1(s) \\ h^2(s) \\ h^3(s) \\ h^4(s) \end{bmatrix} + \begin{bmatrix} v^1 \\ v^2 \\ v^3 \\ v^4 \end{bmatrix} \tag{32}$$

where

$$\begin{bmatrix} h^1(s) \\ h^2(s) \\ h^3(s) \\ h^4(s) \end{bmatrix} := \begin{bmatrix} -\underline{u}_1(s) \\ \underline{u}_3(s) \\ -\underline{\underline{u}}_1(s) \\ \underline{\underline{u}}_3(s) \end{bmatrix} \tag{33}$$

We say that the measurements m are not observable, meaning that there are two different states, s and f, that the measurements m can not distinguish, i.e., h(s) = h(f). This nonobservability is essentially caused by the lack of information about $u_2(s)$ in the model (31).

The vector u(s) is the apparent direction cosines of the observed star with respect to the tracker base reference axes and its second component $u_2(s)$ is either positive or negative $\sqrt{1 - u_1^2(s) - u_3^2(s)}$. Furthermore, the star tracker looks in the positive direction of $u_2(s)$. Hence, $u_2(s) = \sqrt{1 - u_1^2(s) - u_3^2(s)}$ and $u_2(s)$ is completely determined by $u_1(s)$ and $u_3(s)$, which are directly measured in (31). Indeed, the physical system that (32) and (33) try to describe is observable. But the direction of the star tracker is by no means taken into consideration in the model (31) (nor in (30)). We will now illustrate this point with the following example.

Example. Assume that

$$A_{\underline{x}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \underline{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$A_{\underline{\underline{x}}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \qquad \underline{\underline{a}} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$s = A_{xz^I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$f = A_{xz^I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Simple calculations yield

$$\underline{u}(s) = \Lambda_{\underline{x}}{}^{t}{}_{x}{}^{s}\,\underline{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\underline{\underline{u}}(s) = \Lambda_{\underline{\underline{x}}}{}^{t}{}_{x}{}^{s}\,\underline{\underline{a}} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\underline{u}(f) = \Lambda_{\underline{x}}{}^{t}{}_{x}{}^{f}\,\underline{a} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$$

$$\underline{\underline{u}}(f) = \Lambda_{\underline{\underline{x}}}{}^{t}{}_{x}{}^{f}\,\underline{\underline{a}} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$$

Hence,

$$h(s) = \begin{bmatrix} -\underline{u}(s) \\ \underline{u}_3(s) \\ -\underline{u}_1(s) \\ \underline{u}_2(s) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -\underline{u}_1(f) \\ \underline{u}_3(f) \\ -\underline{\underline{u}}_1(f) \\ \underline{\underline{u}}_3(f) \end{bmatrix} = h(f)$$

We note that this example is also valid for the model (30) for which $h(s) = [-\underline{u}_1(s)/\underline{u}_2(s), \underline{u}_3(s)/\underline{u}_2(s), -\underline{\underline{u}}_1(s)/\underline{\underline{u}}_2(s), \underline{\underline{u}}_3(s)/\underline{\underline{u}}_2(s)]^T$.

As the star trackers look in the positive u_2 direction, the direction cosines u(f), of which the second component is negative, can not be those of the observed star. They only form a pseudo-image of the observed star, which the measurements (32) and (33) can not distinguish from the real image of the same star.

VII. Remedying the Nonobservability

We recall that the conditional density $p(m_k|s_k)$ is used in calculating $p(s_k|m^k)$ and that

$$p(m_k|s_k)=(2\pi)^{-\frac{r}{2}}(\det R_k)^{-\frac{1}{2}}\exp\left[-\frac{1}{2}\sum_{i,j=1}^{r} R_k^{ij}(m_k^i-h^i(s_k,k))(m_k^j-h^j(s_k,k))\right] \qquad (15)$$

where r is the dimension of the measurement vector m. From this equation, it is clear that $p(m_k|s_k)$ takes on the same value at the state s and at the pseudo-state f resulting respectively from the images u(s) and the pseudo-images u(f) of the observed stars, since h(f) = h(s).

In order to eliminate such effect of the star pseudo-images, a new approach, which we call the pseudo-measurement technique, was developed. The idea is pretty simple. Supposing that $\underline{u}_2$ and $\underline{\underline{u}}_2$ were directly measured and (32) and (33) became

$$\begin{bmatrix} m^1 \\ m^2 \\ m^3 \\ m^4 \\ m^5 \\ m^6 \end{bmatrix} = m = h(s) + v = \begin{bmatrix} h^1(s) \\ h^2(s) \\ h^3(s) \\ h^4(s) \\ h^5(s) \\ h^6(s) \end{bmatrix} + \begin{bmatrix} v^1 \\ v^2 \\ v^3 \\ v^4 \\ v^5 \\ v^6 \end{bmatrix} \qquad (41)$$

and

$$\begin{bmatrix} h^1(s) \\ h^2(s) \\ h^3(s) \\ h^4(s) \\ h^5(s) \\ h^6(s) \end{bmatrix} := \begin{bmatrix} -\underline{u}_1(s) \\ \underline{u}_3(s) \\ -\underline{\underline{u}}_1(s) \\ \underline{\underline{u}}_3(s) \\ \underline{u}_2(s) \\ \underline{\underline{u}}_2(s) \end{bmatrix} \tag{42}$$

the equation (15) would remain unchanged with r = 6 instead of 4. Of course, m_k^5 and m_k^6 are not really available. However, if the estimate $\hat{s}_{k-1}$ of s_{k-1} is good, then $\underline{u}_2(w_{k-1} \circ \hat{s}_{k-1})$ and $\underline{\underline{u}}_2(w_{k-1} \circ \hat{s}_{k-1})$ should be equally good estimates of m_k^5 and m_k^6. It seems only natural to use $\underline{u}_2(w_{k-1} \circ \hat{s}_{k-1})$ and $\underline{\underline{u}}_2(w_{k-1} \circ \hat{s}_{k-1})$ as measurements of $\underline{u}_2(s_k)$ and $\underline{\underline{u}}_2(s_k)$, respectively.

Now setting

$$\begin{aligned} v^5 &= \underline{u}_2(w_{k-1} \circ \hat{s}_{k-1}) - \underline{u}_2(s_k) \\ v^6 &= \underline{\underline{u}}_2(w_{k-1} \circ \hat{s}_{k-1}) - \underline{\underline{u}}_2(s_k) \\ m^5 &= \underline{u}_2(w_{k-1} \circ \hat{s}_{k-1}) \\ m^6 &= \underline{\underline{u}}_2(w_{k-1} \circ \hat{s}_{k-1}) \end{aligned} \tag{43}$$

we retrieve exactly (42) and (43). Here, the question is whether we still have (15) with r=6. This obviously depends on the conditional sampling distributions of v^5 and v^6 in (44) given the state s_k at time k which unfortunately remains unclear as of today. We note only that $\hat{s}_{k-1}$ is a measurable function of $m^{k-1} = \{m_1, \ldots, m_{k-1}\}$ and hence $v^5(k)$ and $v^6(k)$ are conditionally independent of $v^i(k)$, i=1, ..., 4.

We recall that the nonobservability problem arises from the missing sign of u_2. Perhaps we do not need all that much information about u_2 as might be possible to obtain from a precise expression of the conditional sampling distributions of v^5 and v^6, as long as the effect of the pseudo-images of the observed stars can be eliminated without affecting (15) with r=4. This can indeed be achieved by assuming that the conditional sampling distributions of v^5 and v^6 are normal. First of all, the equation (15) is valid again with r=6 under this assumption. Secondly, by setting R^{55} and R^{66} equal to 4, we assign a probability of 95% to a positive value of u_2, leaving the equation (15) with r=4 relatively intact. Thirdly, in view of the central limit theorem, the assumption may not be hard to swallow after all.

VIII. An Estimation Error Criterion and the Optimal Estimate

In order to define an error criterion for orientation estimation, it is necessary to have a measure of the distance between two orientations. We will first describe such a measure, using quaternions. We recall that a rotation about an axis in the direction of a unit vector $[\ell,m,n]^T$ through an angle ϕ is represented by the (unit) quaternion

$$q = [q_1,q_2,q_3,q_4]^T = [\ell\sin\frac{\phi}{2}, m\sin\frac{\phi}{2}, n\sin\frac{\phi}{2}, \cos\frac{\phi}{2}]^T$$

Given two orientations, the minimal angle in radians required to bring one into the other is a natural measure of distance between them and defines a Riemannian metric on SO(3). If the orientations are represented by the quaternions, q and p, and the minimal angle is denoted by $\rho(q,p)$, then we have $q^Tp = \cos\frac{1}{2}\rho(q,p)$. As $\frac{1}{2}(1 - \cos\rho)$ is a monotone increasing function of ρ, a measure of distance between p and q can be defined to be $\frac{1}{2}(1 - \cos\rho(q,p)) = 1 - (q^Tp)^2$. It can be shown that if the orientations, q and p, are described by the 3x3-dimensional orthogonal matrices, Q and P, then this measure of distance can also be expressed as $\frac{1}{4}(3 - \mathrm{tr}\, PQ^T)$.

We are now ready to define the error criterion for orientation estimation. Let q be a random quaternion and p its estimate. Then a measure of the estimation error is

$$J(q,p) = E(1 - (q^Tp)^2)\,. \tag{44}$$

If the probability distribution of q is given, the estimate p which minimizes J may be obtained from observing that $J(q,p) = 1 - p^T E(qq^T)p$. It is well-known that the quadratic form $p^T Vp$ of the positive definite matrix $V = E(qq^T)$ is maximized when p is the unit eigenvector associated with the largest eigenvalue λ of V. Moreover, the maximum value is λ. Hence,

$$\begin{aligned}\min_p J(q,p) &= 1 - \hat{q}^T E(qq^T)\hat{q}\\ &= 1 - \lambda\end{aligned}$$

where λ = the maximum eigenvalue of $E(qq^T)$

$\hat{q}$ = the unit eigenvector of $E(qq^T)$ associated with λ .

The probability distributions on SO(3) are expressed in terms of the Euler angles (ϕ,θ,ψ) in (26). The following relationships between the quaternion components and the Euler angles will have to be used to calculate the optimal estimate $\hat{q}$ and its estimation error $1 - \lambda$:

$$q_1 = \sin\frac{\theta}{2}\cos\frac{\phi-\psi}{2}$$

$$q_2 = \sin\frac{\theta}{2}\sin\frac{\phi-\psi}{2}$$

$$q_3 = \cos\frac{\theta}{2}\sin\frac{\phi+\psi}{2}$$

$q_4 = \cos\frac{\theta}{2}$ [illegible] $+\psi$

IX. Simulation Results

Three simulated examples with various system parameters and noise levels are given in this section. They were chosen to test the robustness of the optimal scheme and therefore represent tougher working conditions than the real ones. The measurement noise, v^i in (42), is set at two different levels to illustrate how the noise level can affect the estimator performance. The variance, $1/R^{ii}$, of v^i is 1/36 for A part of the examples, and is 10^{-4} for B part of the examples. The reader is rerferred to the Module II report for the specification of all the system parameters.

The four graphs of the A part of the $i^{\underline{th}}$ example are, respectively,

GRAPH (i, A, 1):

——————— the error, ERH1, of the optimal estimate obtained by local integration

- - - - - - - the error, ERH2, of the maximum likelihood estimate

GRAPH (i, A, 2):

The same as GRAPH (i, A, 1) except that the vertical scale is changed and the points at k = 1 and 2 are removed to magnify the vertical variations.

GRAPH (i, A, 3):

——————— the difference, ERH1 - ERH2, of the errors

- - - - - - - the distance between the maximum likelihood estimate and the optimal estimate obtained by local integration

GRAPH (i, A, 4):

The same as GRAPH (i, A, 3) except that the vertical scale is changed and the points at k = 1 and 2 are removed to magnify the vertical variations.

The symbols to be used to specify the system parameters and noise levels are defined as follows:

1. The attitude propagation equation is

$$s_{k+1}(\phi_{k+1}, \theta_{k+1}, \psi_{k+1}) = w_k(\alpha_k, \beta_k, \gamma_k) \circ s_k(\phi_k, \theta_k, \psi_k) \qquad (3)$$

2. The initial attitude at k=0 is represented by a quaternion $s_0 = [q_1, q_2, q_3, q_4]^T$.

3. The initial distribution of s_0 has a rotational normal density

$$p(s_0) = c \exp [a \, (\sum_{i=1}^{4} b_i q_i)^2]$$

4. The Euler angles α_k, β_k, γ_k are obtained from multiplying the constants c_1, c_2, c_3, resp., by pseudo-random numbers from the uniform distribution on the interval [0, 1].

5. The star tracker measurement and the pseudo-measurement equations are

$$m_k^i = h^i(s_k, k) + v_k^i, \; i=1, \ldots, 6 \qquad (42)$$

where

$$P(v) = (2\pi)^{-3} \left(\sum_{i=1}^{6} R^{ii} \right)^{-\frac{1}{2}} \exp \left(-\frac{1}{2} \sum_{i=1}^{6} R^{ii}(v^i)^2 \right)$$

6. The apparent direction cosines u with respect to a star tracker and those a with respect to the earth are related by

$$u(k) = A_{x^t x} A_{xz^I}(k) \, a(k) \qquad (29)$$

EXAMPLE 1.

$$s_0 = \begin{bmatrix} \sin\frac{\pi}{6}\ (\sin\frac{\pi}{3}\cos\frac{\pi}{4}) \\ \sin\frac{\pi}{6}\ (\sin\frac{\pi}{3}\sin\frac{\pi}{4}) \\ \sin\frac{\pi}{6}(\cos\frac{\pi}{3}) \\ \cos\frac{\pi}{6} \end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} \sin\frac{5\pi}{36}(\sin\frac{2\pi}{5}\ \cos\frac{5\pi}{16}) \\ \sin\frac{5\pi}{36}(\sin\frac{2\pi}{5}\ \sin\frac{5\pi}{16}) \\ \sin\frac{5\pi}{36}\cos\frac{2\pi}{5} \\ \cos\frac{5\pi}{36} \end{bmatrix}$$

$$a = 2.0$$

$$[c_1, c_2, c_3] = [\frac{\pi}{5}, \frac{\pi}{10}, \frac{\pi}{5}]$$

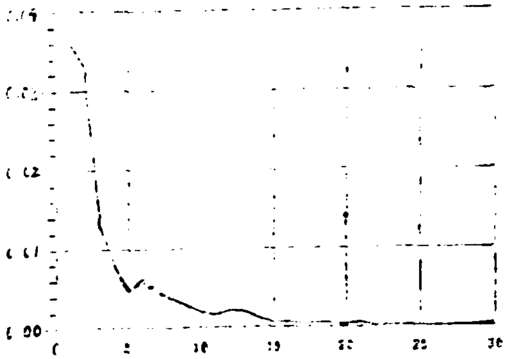
$$R^{ii} = 36.0, \quad i=1, \ldots, 4$$

$$R^{ii} = 4.0\ , \quad i=5 \text{ and } 6$$

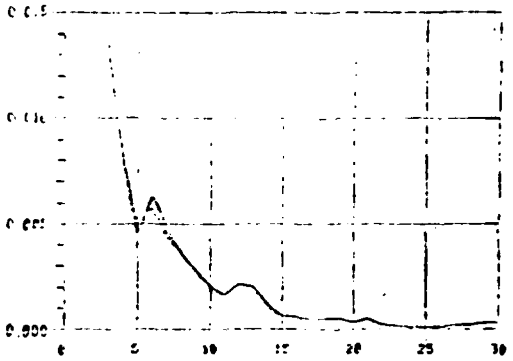
$$A_{\underline{x}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$A_{\underline{\underline{x}}}{}^{t}{}_{x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

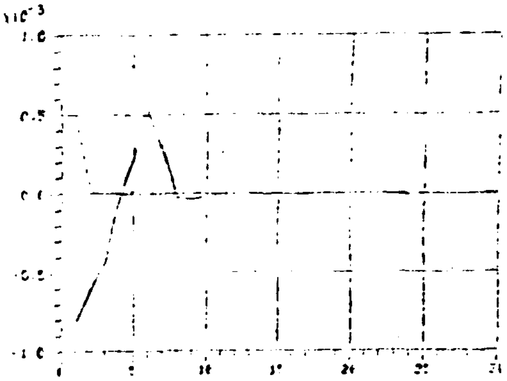
PART A



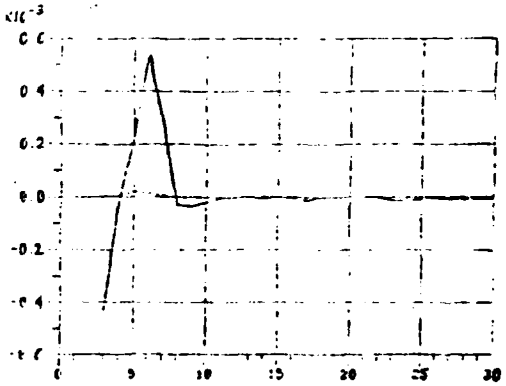
GRAPH (1, A, 1)



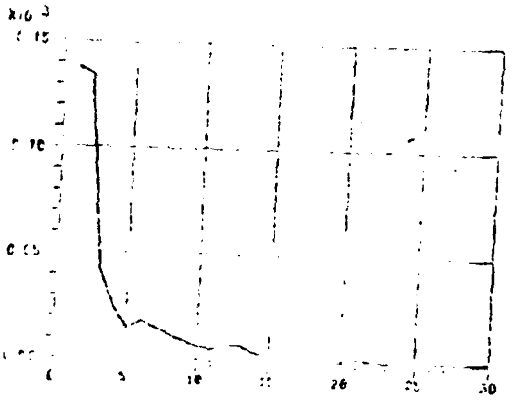
GRAPH (1, A, 2)



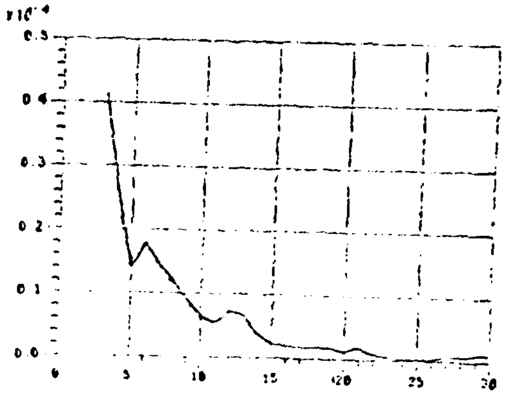
GRAPH (1, A, 3)



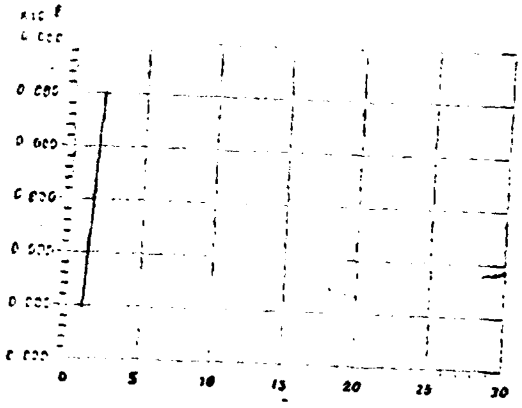
GRAPH (1, A, 4)



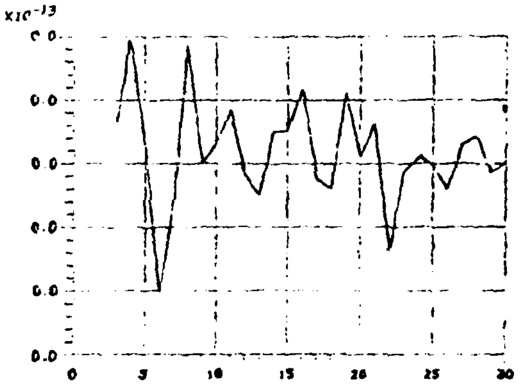
GRAPH (1, B, 1)



GRAPH (1, B, 2)



GRAPH (1, B, 3)



GRAPH (1, B, 4)

EXAMPLE 2.

$$s_0 = \begin{bmatrix} \sin\frac{\pi}{3}\ (\sin\frac{\pi}{2}\cos\frac{\pi}{3}) \\ \sin\frac{\pi}{3}\ (\sin\frac{\pi}{2}\sin\frac{\pi}{3}) \\ \sin\frac{\pi}{3}(\cos\frac{\pi}{2}) \\ \cos\frac{\pi}{3} \end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} \sin\frac{\pi}{4}\ (\sin\frac{\pi}{3}\ \cos\frac{\pi}{2}) \\ \sin\frac{\pi}{4}\ (\sin\frac{\pi}{3}\ \sin\frac{\pi}{2}) \\ \sin\frac{\pi}{4}(\cos\frac{\pi}{3}) \\ \cos\frac{\pi}{4} \end{bmatrix}$$

a=2.0

$[c_1, c_2, c_3] = [\frac{2\pi}{5}, \frac{2\pi}{5}, \frac{2\pi}{5}]$

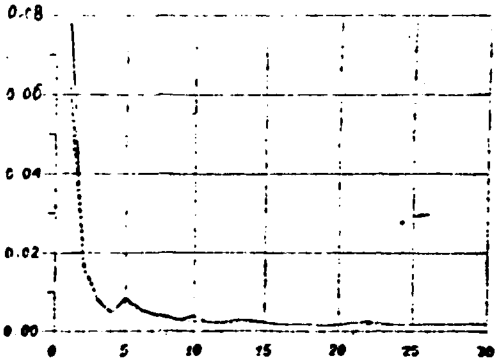
$R^{ii} = 36.0, \quad i = 1, \ldots, 4$

$R^{ji} = 4.0, \quad i = 5 \text{ and } 6$

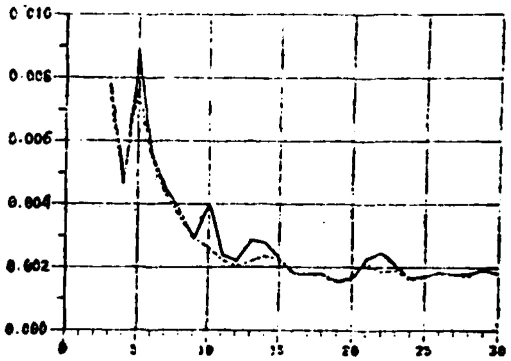
$$A_{\underline{x}^t x} = \begin{bmatrix} \frac{1}{2} & 0 & -\frac{\sqrt{3}}{2} \\ 0 & 1 & 0 \\ \frac{\sqrt{3}}{2} & 0 & \frac{1}{2} \end{bmatrix}$$

$$A_{\underline{\underline{x}}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

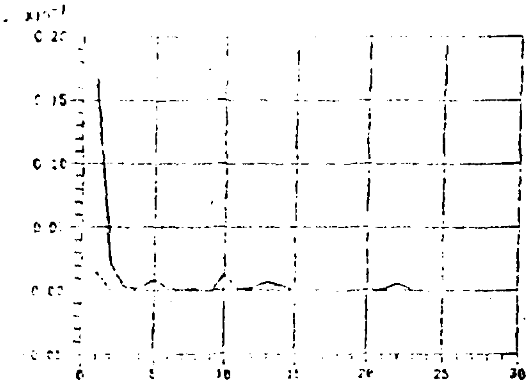
PART A



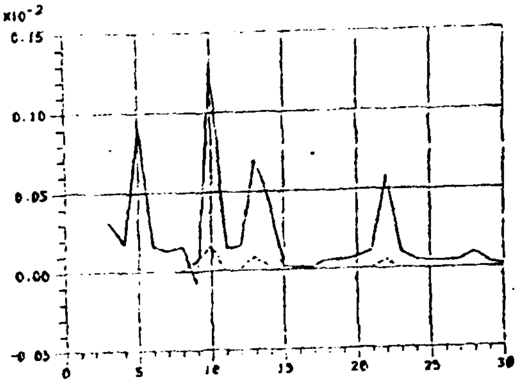
GRAPH (2, A, 1)



GRAPH (2, A, 2)

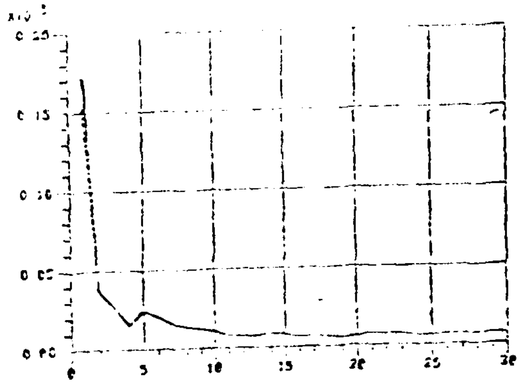


GRAPH (2, A, 3)

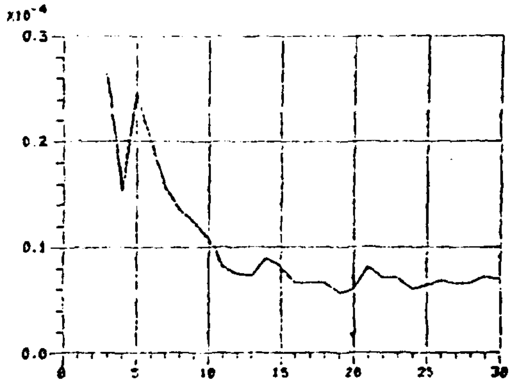


GRAPH (2, A, 4)

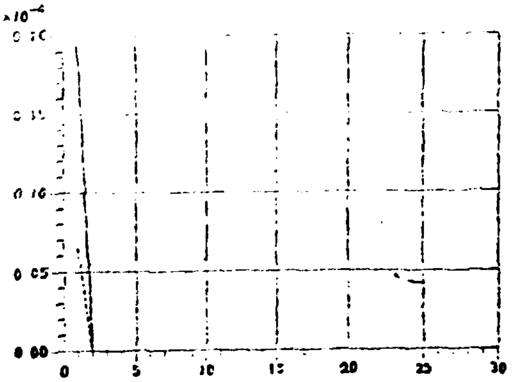
PART B



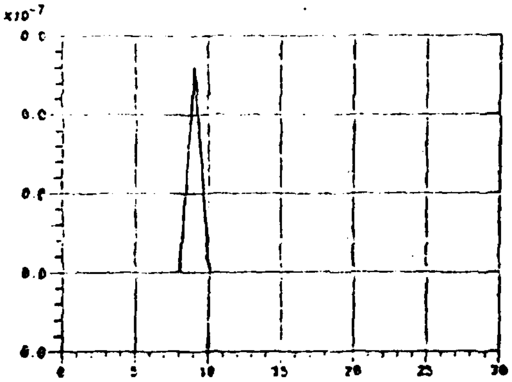
GRAPH (2, B, 1)



GRAPH (2, B, 2)



GRAPH (2, B, 3)



GRAPH (2, B, 4)

EXAMPLE 3

$$s_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$a = 2.0$$

$$[c_1, c_2, c_3] = [\frac{2\pi}{5}, \frac{2\pi}{5}, \frac{2\pi}{5}]$$

$$R^{ii} = 36.0, \quad i = 1, \ldots, 4$$

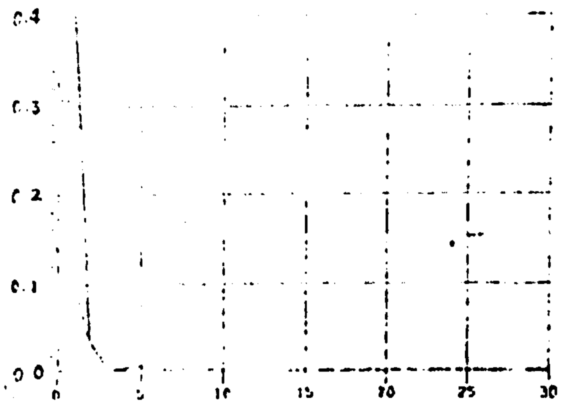
$$R^{ii} = 4.0, \quad i = 5 \text{ and } 6$$

$$A_{\underline{x}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

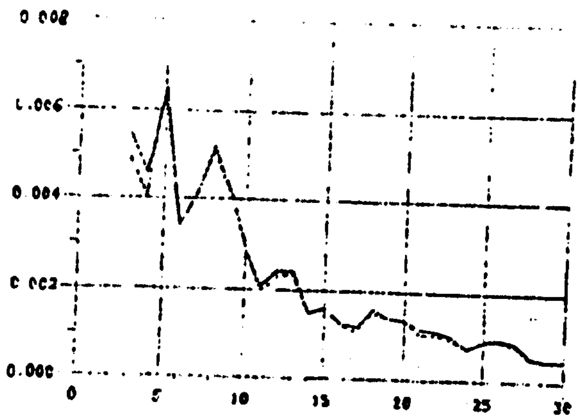
$$A_{\underline{\underline{x}}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

Remark. Let us recall the nonobservability example given in Section II. The pseudo-state of the real state s_0 is exactly $[b_1, b_2, b_3, b_4]$, which is chosen to be the mode of the initial density of s_0. The pseudo-measurement technique thus picks wrongly the pseudo-state at the first time-point. This example was in fact deliverately designed to work against the pseudo-measurement technique. The simulation results to be depicted in the following graphs indicate without doubt that the pseudo-measurement technique has unbelievable self-correcting power and switches to the real images of the stars very quickly (at the second time-point in this example).

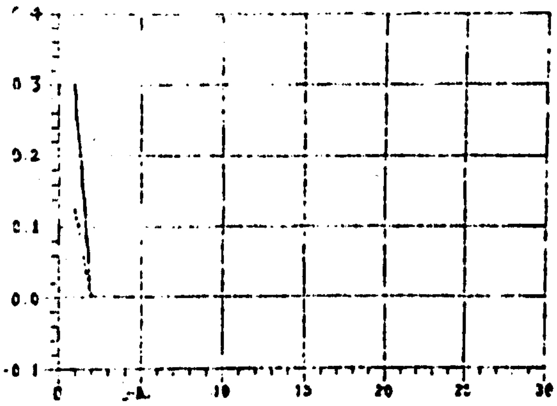
PART A



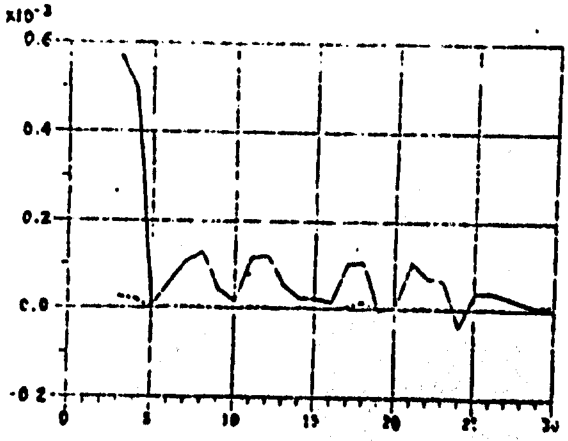
GRAPH (3, A, 1)



GRAPH (3, A, 2)

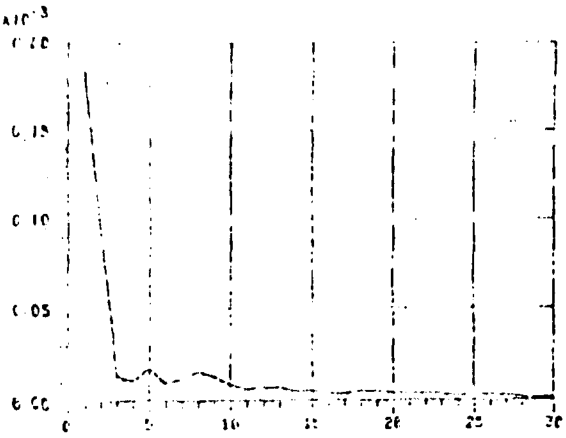


GRAPH (3, A, 3)

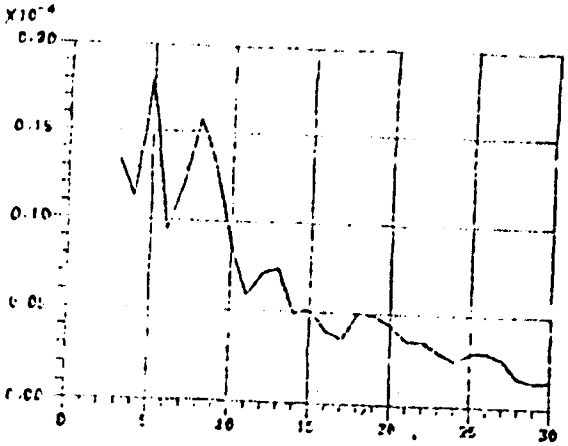


GRAPH (3, A, 4)

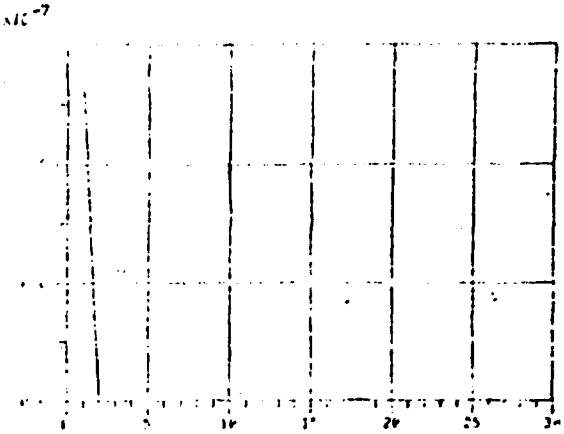
PART B



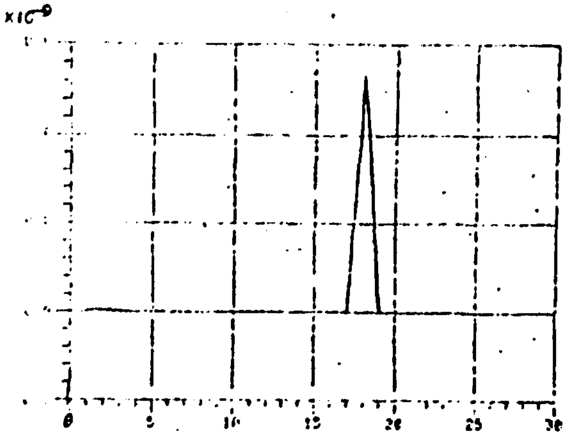
GRAPH (3, B, 1)



GRAPH (3, B, 2)



GRAPH (3, B, 3)



GRAPH (3, B, 4)

III. Comparison Between the Maximum Likelihood Estimates and the K-B Estimates

We note again that it takes much more CPU time to compute the optimal estimates even by local integration and, based on the simulation results, the local integration estimates are not much better than the maximum likelihood estimates. Therefore, a comparison was made between the K-B estimation and the maximum likelihood estimation instead of the optimal estimation.

The comparison was conducted by E. J. Lefferts of the GSFC/NASA. A. N. Mansfield of the CSTA generated sequences of 33 star tracker observations. The average body angular velocity was provided every one third of a second, and the tracker observation was taken every two minutes. The standard deviation of the tracker measurement noises is 20 arcseconds.

For such a low noise level, it is known that linearization is a very good approximation and the extened K-B filter is expected to be near optimal. Indeed, our comparison results confirmed these long-standing conjectures. Two typical examples were included in the following. We note that the maximum likelihood estimates are almost always better and converge faster than the K-B estimates. However they are close, especially in the steady states.

The system parameters and the noise statistics for the two comparison runs to be reported in the following will now be given, using the symbols established in Section IV of the Module II report:

$a = 2 * (180/2\pi)^2$

$R^{ii} = \left[20 * (1/3600) * (\pi/180)\right]^{-2}$, i=1, 2, 3, 4

$R^{ii} = 0$, i=5, 6

$$A_{\underline{x}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$${}_{\underline{\underline{x}}^t x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

The four graphs of the i$\underline{\text{th}}$ example are, respectively,

GRAPH (i, 1):

———————— the angular error in arcseconds of the K-B estimate

- - - - - - - the angular error in arcseconds of the maximum likelihood estimate

GRAPH (i, 2):

The same as GRAPH (i, 1) except that the vertical scale is changed and the points at k=1, ..., 4 are removed to magnify the vertical variations.

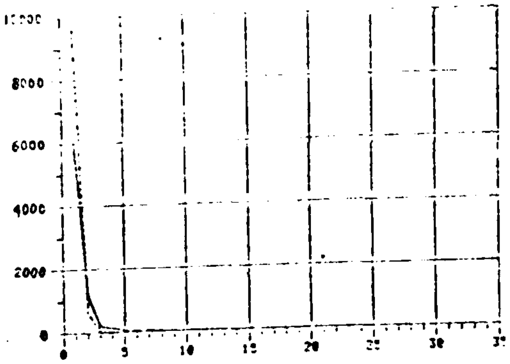
GRAPH (i, 3):

- - - - - - - - - the angular error of the K-B estimate minus that of the maximum likelihood estimate

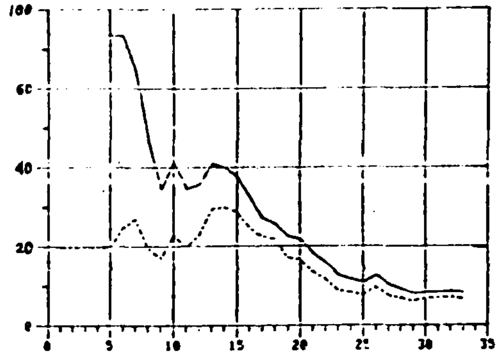
GRAPH (i, 4):

The same as GRAPH (i, 3) except that the vertical scale is changed and the points at k=1, ..., 4 are removed to magnify the vertical variations.

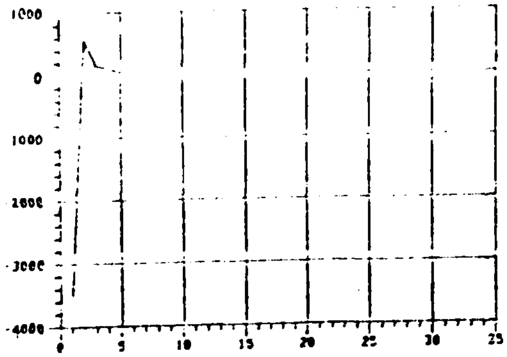
EXAMPLE 4.



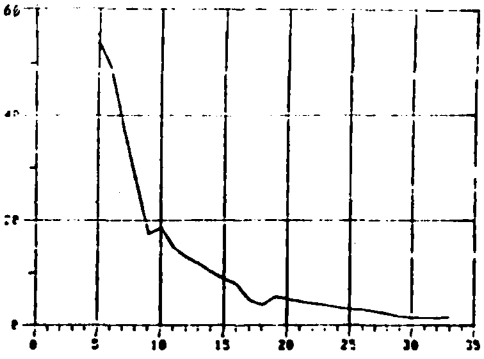
GRAPH (4, 1)



GRAPH (4, 2)

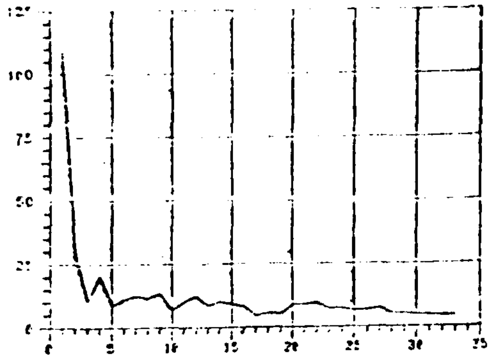


GRAPH (4, 3)

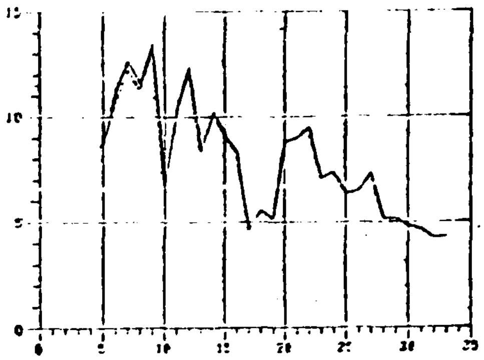


GRAPH (4, 4)

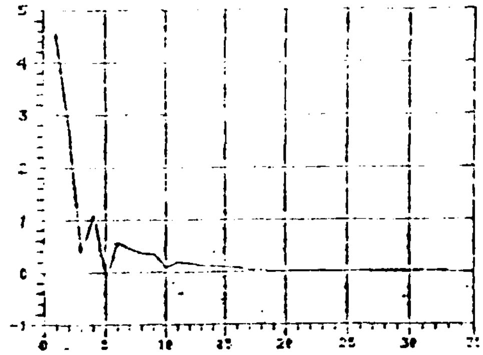
EXAMPLE 5.



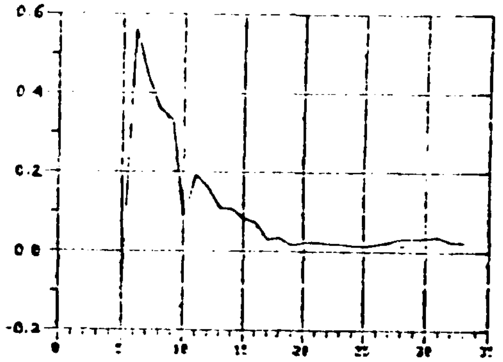
GRAPH (5, 1)



GRAPH (5, 2)

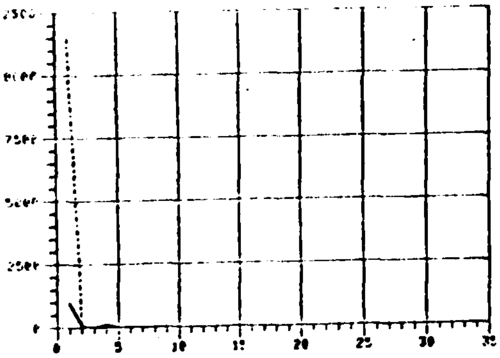


GRAPH (5, 3)

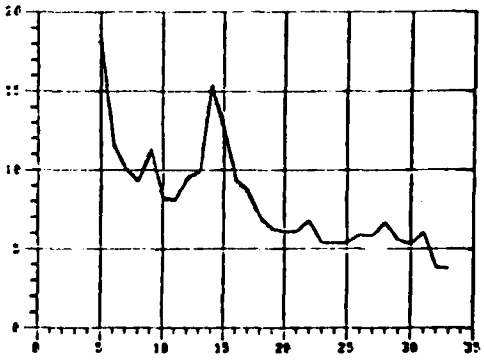


GRAPH (5, 4)

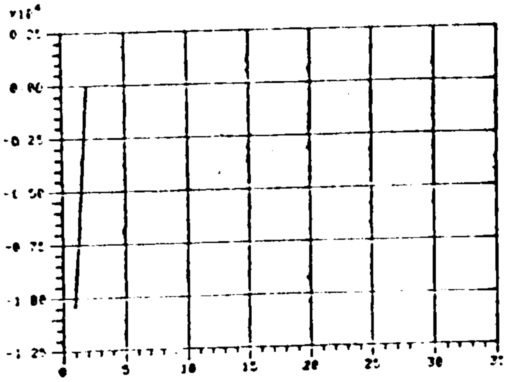
EXAMPLE 6.



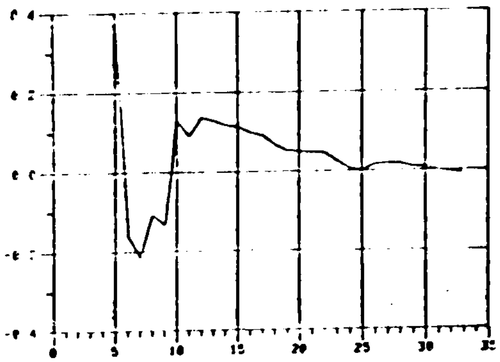
GRAPH (6, 1)



GRAPH (6, 2)



GRAPH (6, 3)



GRAPH (6, 4)

ACKNOWLEDGMENTS

The author would like to acknowledge with thanks Mr. Eugene J. Lefferts of GSFC/NASA for many valuable discussions and suggestions and Dr. Arthur Mansfield of CSTA for generating measurement data. Special thanks go to Dr. Masahiro Nishihama for his excellent programming job. This work would not have been possible without their contributions.

REFERENCES

[1]. HEAO-75-460-177, "Attitude Determination Software for HEAO", R. L. Farrenkopf, dated 10 April 1975.

[2]. R. L. Farrenkopf, R. P. Iwens, and F. J. Belsky, "Simulation Software for The Precision Pointing Control System, I. The Attitude Reference Simulation", TRW Systems Technical Report dated 21 April 1970.

[3]. J. T. Lo, "Optimal Estimation and Detection for Rotational Processes", Presented at the 1976 NASA Symposium on Flight Mechanics/Estimation Theory, Goddard Space Flight Center, October 1976.

[4]. J. T. Lo and L. R. Eshleman, "Exponential Fourier Densities on SO(3) and Optimal Estimation and Detection for Rotational Processes", UMBC Mathematics Research Report No. 76-9, Department of Mathematics, UMBC, July 1976, also presented as an invited paper at the 1976 IEEE Symposium on Decision and Control, Clearwater Beach, Florida, December 1976.

[5]. J. D. Talman, Special Functions: A Group Theoretic Approach, W. A. Benjamin, Inc., New York, 1968.

APPENDIX

MØ5S03

Main Program

Purpose:

To obtain the optimal estimate $\hat{s}_k$, maximum liklihood estimate $\tilde{s}_k$, of the satellite attitude s_k satisfying

$$s_k(\phi_k, \theta_k, \psi_k) = w_k(\alpha_k, \beta_k, \gamma_k) \circ s_{k-1}(\phi_{k-1}, \theta_{k-1}, \psi_{k-1}),$$

using the measurements $\{m_1, m_2, \ldots m_k\}$ acquired from two star trackers, which are described by

$$m_k = h(s_k, K) + v_k$$

Common variables:

/BLK1/	N, NZ, MT
/BLK2/	PAI2, HØ, H, D, AXX
/BLK3/	AD, AA, BB, WØ, OBN, PØ, QØ, P1, Q1,
/BLK4/	AK, QS, DIR, QW, RR, SINE, COSN, QSN, QH1, QH2, QHØ, ERRN, ERH1, ERH2, ER12

Input:

PAI2	2π	
N	number of subintervals in $[0, 2\pi]$	
NZ	number of subintervals in $[0, 2\pi]$ for the first and third Euler angles ϕ, ψ	
MT	number of subinteryals in $[0, \pi]$ for the second Euler angle θ	
HØ	mesh for integration	$(= 2\pi/N)$
H	mesh for integration	$(= 2\pi/NZ)$
D	mesh for integration	$(= \pi/MT)$

AXX 2x3x3 dimensional array representing each orientation of the star tracker base reference axes with respect to the satellite base reference axes.

AD 2x3 - dimensional array representing the apparent direction cosines of a star observed by each of the two star trackers with respect to the earth-centered inertial set of coordinate system.

WØ satellite attitude propagation, Euler angles, 3-dim. array.

QW quaternion form of WØ, 4-dim. array.

PØ, QØ coefficients of harmonics in $P(s_{k-1}|m^{k-1})$, which will be updated for $P(s_k|m^k)$ to P1, Q1, 2x5x5-dim. arrays.

AK parameter of the rotational normal distribution of the initial satellite attitude S_0.

AA, BB coefficients of $\{ReD^1_{mn}(s_k)\}$, $\{ImD^1_{mn}(s_k)\}$ in 6-dim. function $h(s_k,K)$, 6x3x3-dim. arrays.

RR diagonal elements of the covariance matrix for the normal distribution of the observation noise, 6-dim. array.

SINE, COSN table of sin ϕ, cos ϕ, 33-dim. arrays.

SDD1, SDD2 $\{d^1_{mn}(\theta)\}$, $\{d^2_{mn}(\theta)\}$ for $ReD^\ell_{mn}(\phi,\theta,\psi)$, $ImD^\ell_{mn}(\phi,\theta\ \psi)$ for $\ell=1,2$, 3x3x17 and 5x5x17-dim. arrays.

QS attitude s_{k-1}, 4-dim. array.

QHØ (k-1)th stage maximum liklihood estimate of s_{k-1} to generate a predicted estimate of s_k, 4-dim. array.

IXØ,IYØ initial random integers which create measurement noise, satellite attitude propagation, and AD.

Output

OBN star tracker measurements, 6-dim. array.

P1,Q1 updated coefficients of the harmonics in the conditional density $P(s_k|m^k)$, 2x5x5-dim. arrays, input for the (k+1)th stage.

QS1 attitude s_k, 4-dim. array, renamed as QS.

QSP predicted estimate of s_k, generated by WØ and QHØ, 4-dim. array.

SH1 estimate $\hat{s}_k$ of the satellite attitude s_k, 4-dim. array.

QH1 sequence of $\{\hat{s}_k\}$, 4x50-dim. array.

SH2 maximum likelihood estimate $\hat{\hat{s}}_k$ of s_k.

QH2 sequence of $\{\hat{\hat{s}}_k\}$, 4x50-dim. array.

ER1 estimation error of the attitude estimate $\hat{s}_k$.

ERRN sequence of estimation errors, 50-dim. array.

ER2 distance between s_k and $\hat{s}_k$, $\rho(s_k, \hat{s}_k)$.

ERH1 sequence of $\{\rho(s_k, \hat{s}_i)\}$, 50-dim. array.

ER3 distance between s_k and $\hat{\hat{s}}_k$, $\rho(s_k, \hat{\hat{s}}_k)$.

ERH2 sequence of $\{\rho({}^{c}{}_{k}, \hat{\hat{s}}_k)$, 50-dim. array.

ER4 distance between $\hat{s}_k$ and $\hat{\hat{s}}_t$, $\rho(\hat{s}_k, \hat{\hat{s}}_k)$.

ER12 sequence of $\{\rho(\hat{s}_k, \hat{\hat{s}}_k)\}$, 50-dim. array.

Special Considerations:

1. Parameter AD is generated at each stage k from s_k considering the deviation from the value a in the equation.

$$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = A_{xx} \cdot A_{xz}(s_k)a$$

this is done in 'ADC'.

2. Attitude propagation WØ is generated at each stage in 'NEWØ'.

3. Measurement OBN consists of six components, which are

   $$\mathrm{OBN}(i) = \mathrm{FU}(i, \mathrm{DD}, \mathrm{AA}, \mathrm{BB}) + V(i) \qquad 1 \le i < 4$$
   $$\mathrm{OBN}(i) = \mathrm{FU}(i, \mathrm{DD}, \mathrm{AA}, \mathrm{BB}) \qquad 5 \le i \le 6$$

   For $1 \le i \le 4$, DD represents $\{ \mathrm{Re}D^1_{mn}(s_k) \}$, $\{ \mathrm{Im}D^1_{mn}(s_k)\}$

   For $5 \le i \le 6$, DD represents $\{ \mathrm{Re}D^1_{mn}(\hat{s}_k) \}$, $\{ I_m D^1_{mn}(\hat{s}_k)\}$,

   where $\hat{s}_k = w_{k-1} \circ \overset{*}{s}_{k-1}$, called predicted estimate of s_k.

   In this sense, OBN is called pseudomeasurements.

4. The initial distribution $p(s_o)$ is assumed to be a rotational normal distribution on SO(3) with parameter AK.

5. In order to reduce integration time, 32 subintervals for ϕ, ψ and 16 subintervals for θ are used in the trapezoidal rule, where $0 \le \phi,\psi \le 2\pi$, $0 \le \theta \le \pi$.

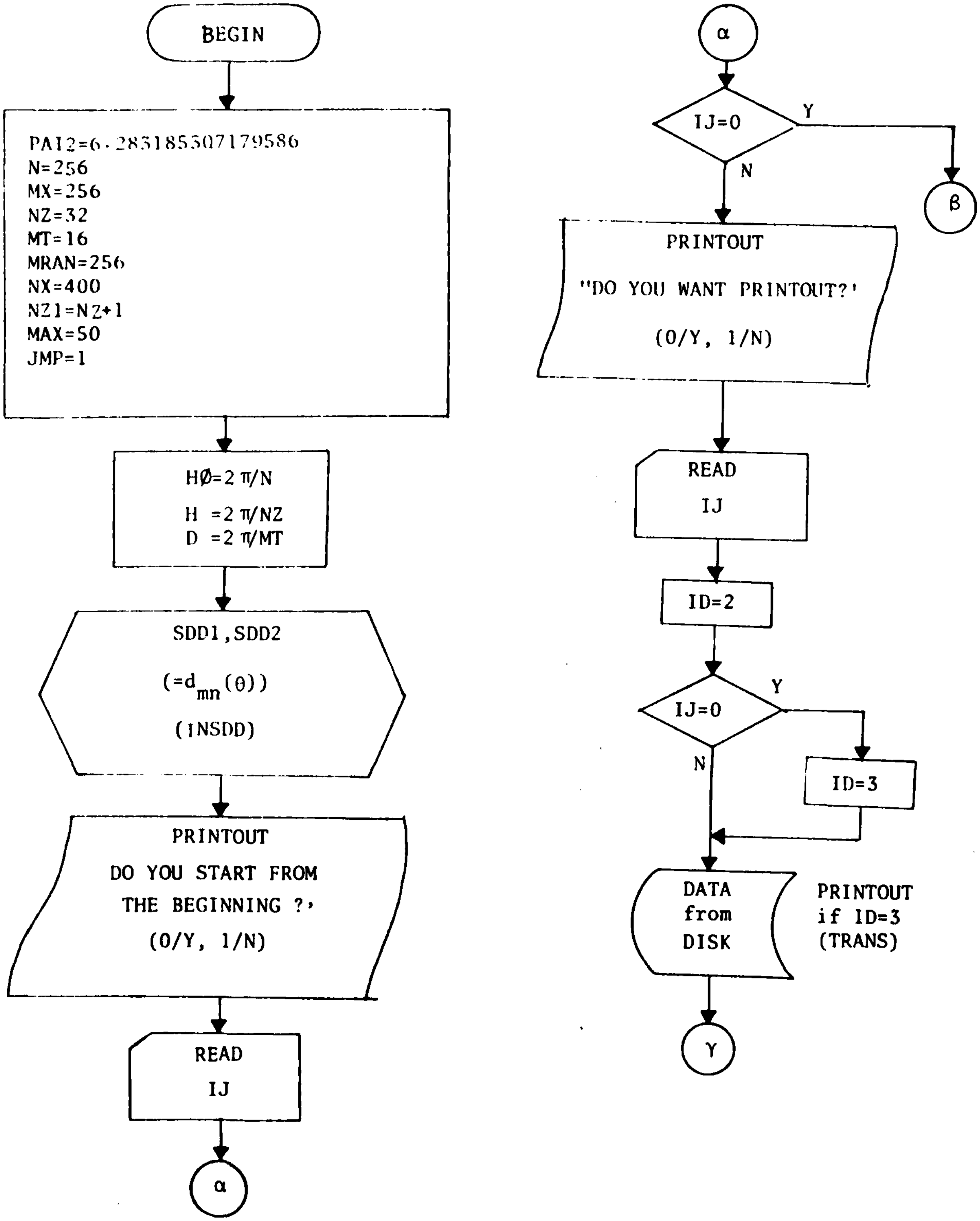
6. In the computation of optimal estimate $\hat{s}_k$ and estimation error, tables of $\sin\phi$, $\cos\phi$, $d^1_{mn}(\theta)$, $d^2_{mn}(\theta)$ are used instead of calling subprograms.

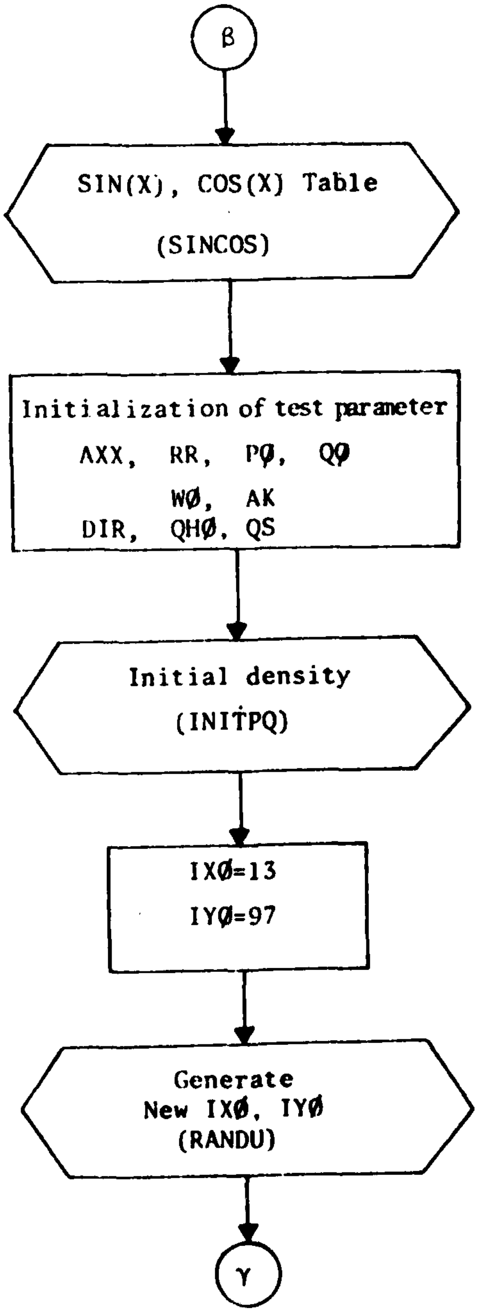
7. Integer KK indicates which stage should be updated and MM shows at (MM-1) stages, the values QSN, QH1, QH2, ERRN, ERH1, ERH2, ER12. Skipping given steps JMP.

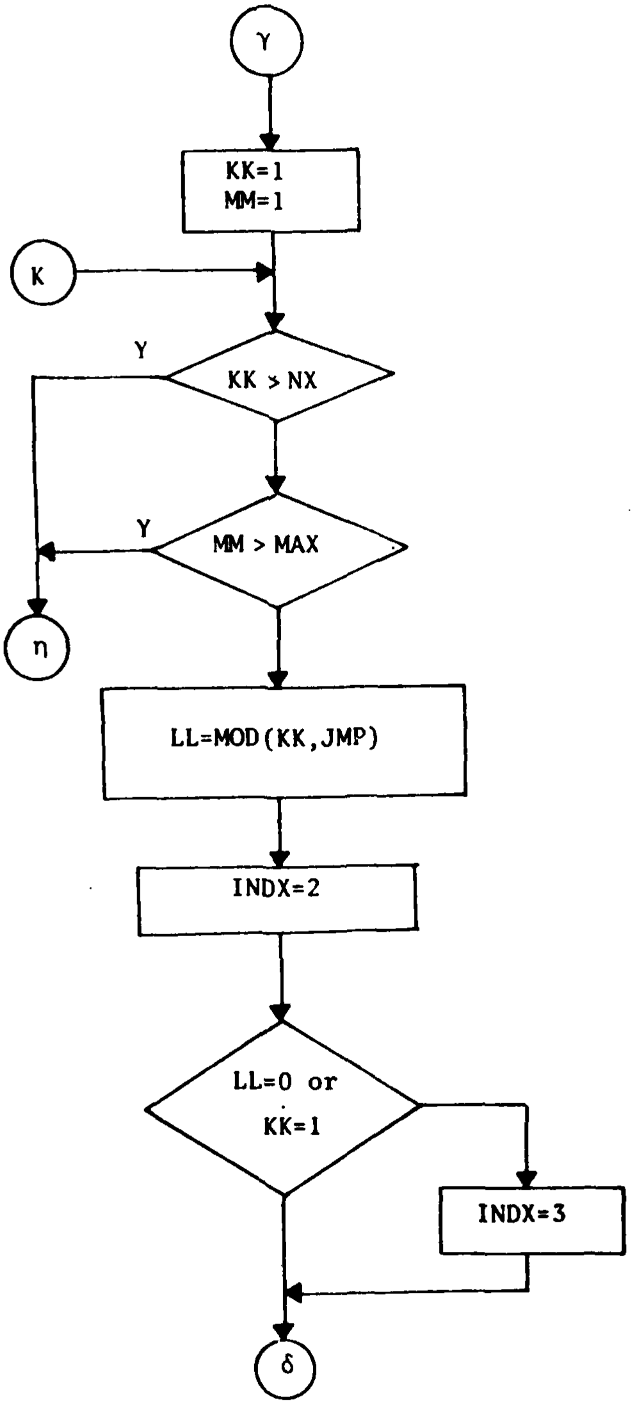
8. At the end of each stage, necessary data are stored in the disk space and will be read at the beginning of RUN by TRANS.

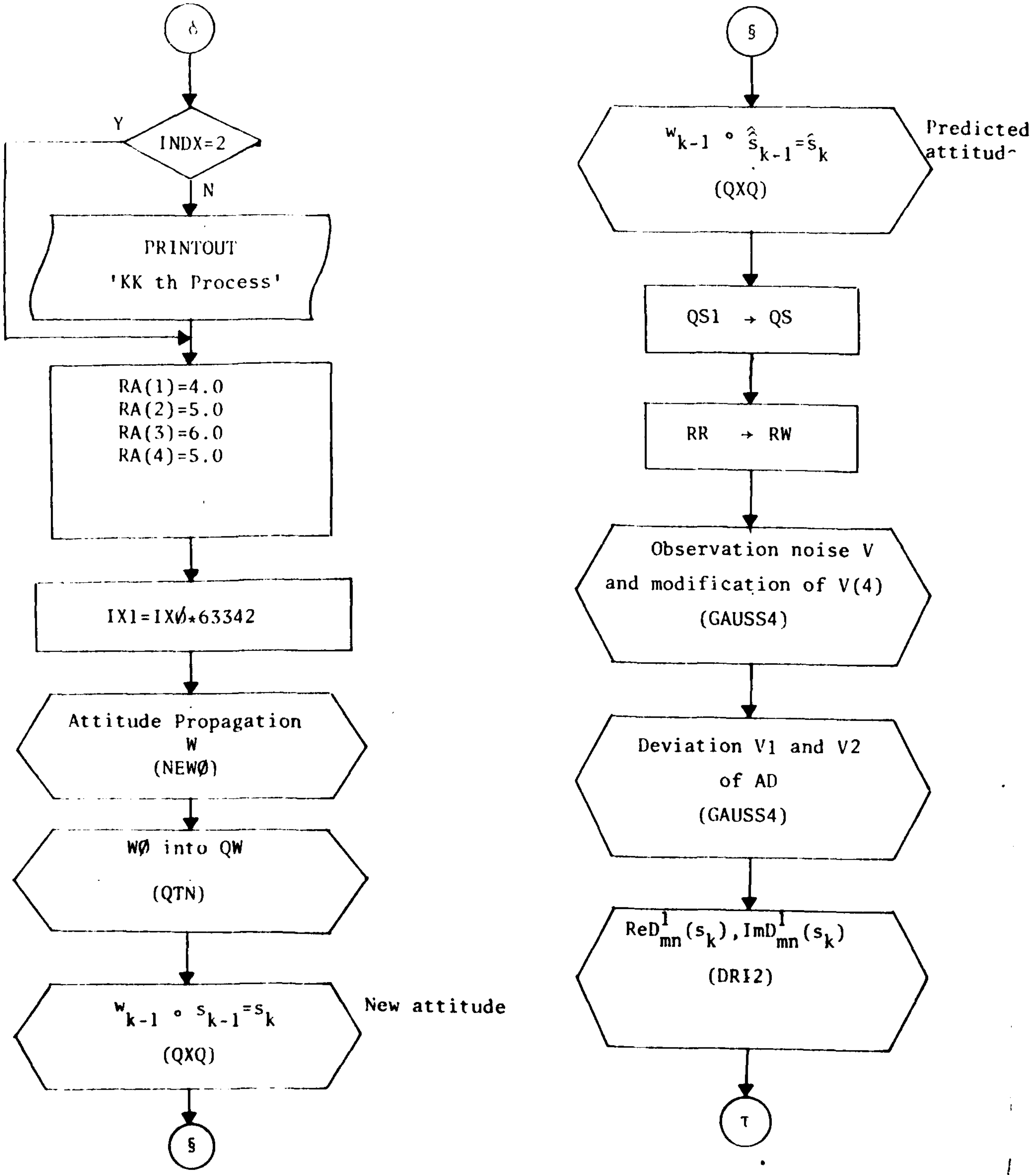
Subprograms Called:

INSDD, TRANS, SINCOS, QTN, INITPQ, RANDU, NEWO, QXQ, GAUSS4, DRI2, ADC, COEFF1, FU, NPAQ, ERR.

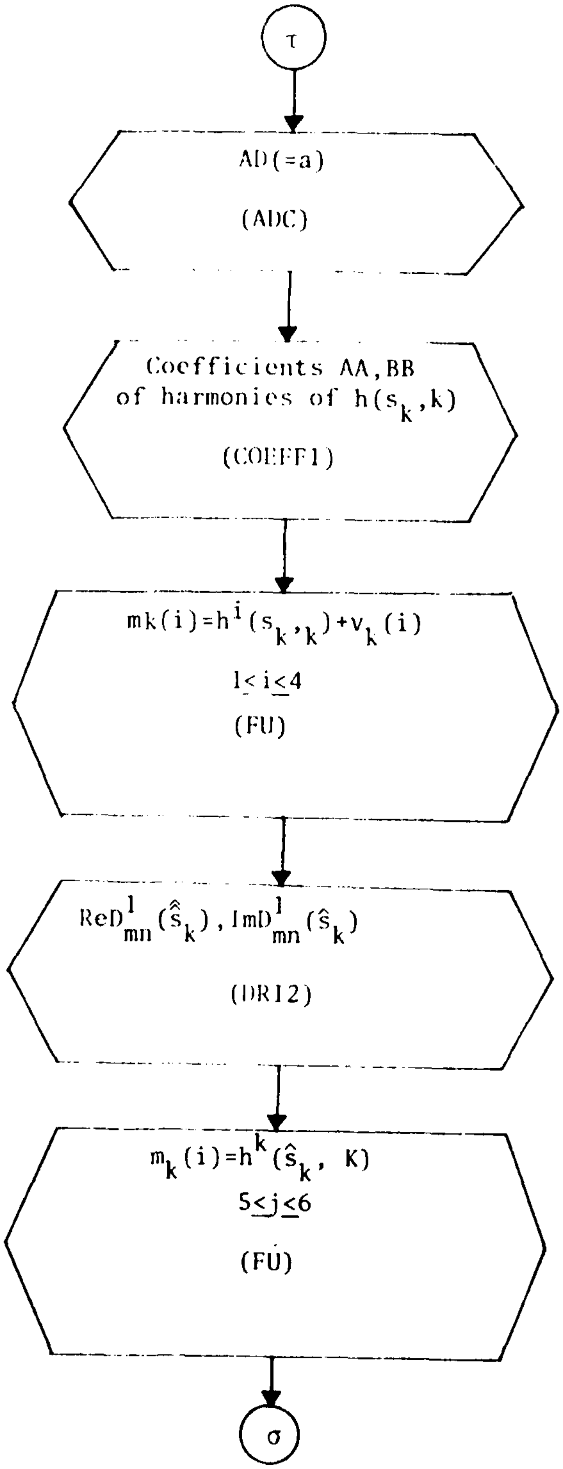

BEGIN
PAI2=6.283185307179586
N=256
MX=256
NZ=32
MT=16
MRAN=256
NX=400
NZ1=NZ+1
MAX=50
JMP=1
HØ=2 π/N
H =2 π/NZ
D =2 π/MT
SDD1,SDD2
(=d_mn(θ))
(INSDD)
PRINTOUT
DO YOU START FROM THE BEGINNING ?'
(0/Y, 1/N)
READ
IJ
α
α
IJ=0
Y
N
β
PRINTOUT
"DO YOU WANT PRINTOUT?'
(0/Y, 1/N)
READ
IJ
ID=2
IJ=0
Y
N
ID=3
DATA
from
DISK
PRINTOUT
if ID=3
(TRANS)
γ

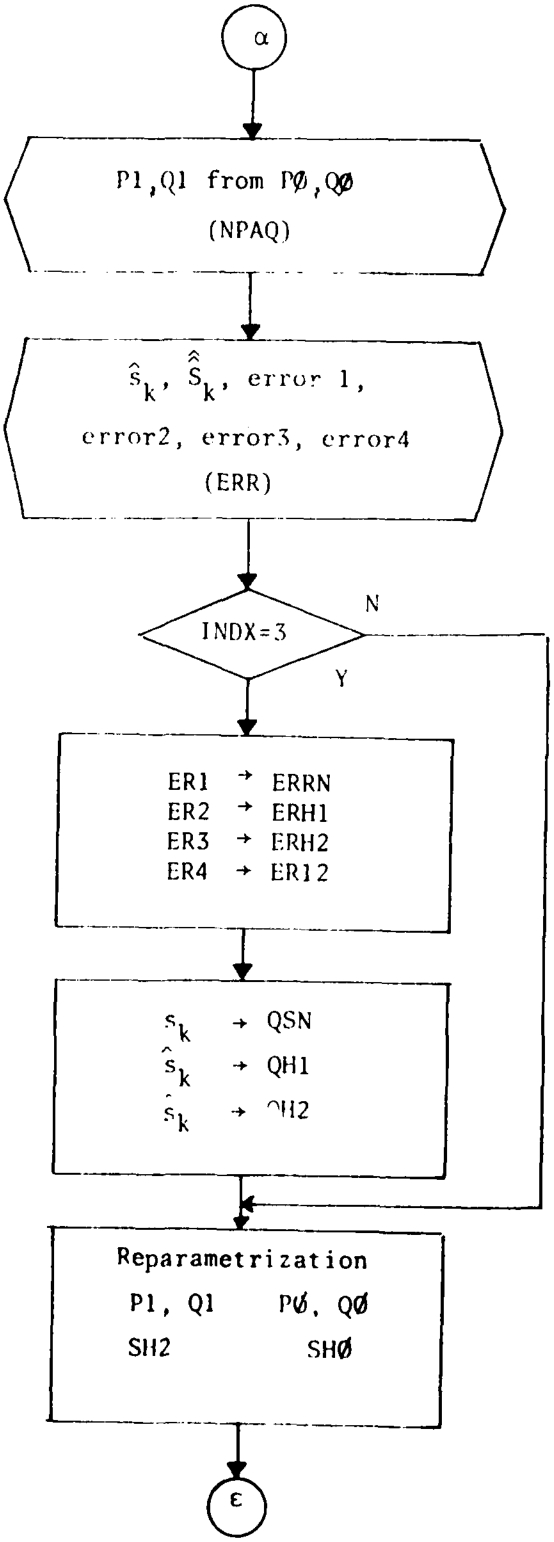

β
SIN(X), COS(X) Table
(SINCOS)
Initialization of test parameter
AXX, RR, PØ, QØ
WØ, AK
DIR, QHØ, QS
Initial density
(INITPQ)
IXØ=13
IYØ=97
Generate
New IXØ, IYØ
(RANDU)
γ

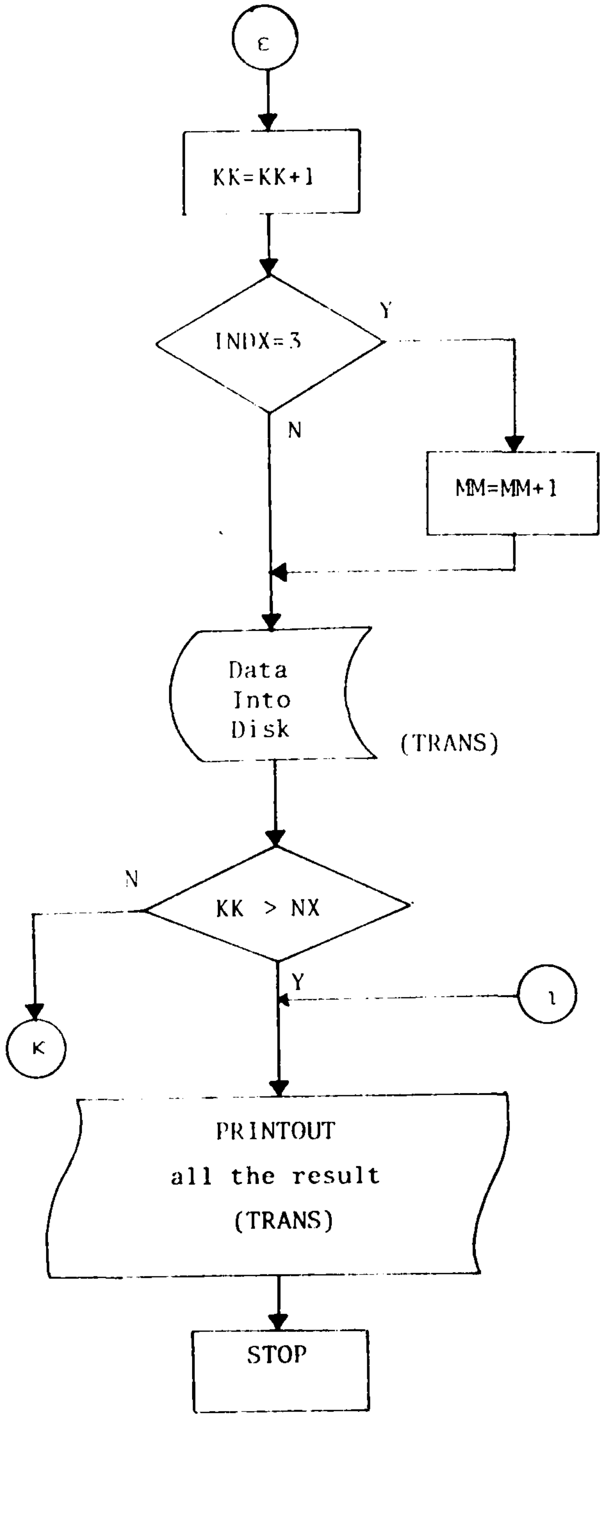

γ
KK=1
MM=1
K
Y
KK > NX
Y
MM > MAX
η
LL=MOD(KK,JMP)
INDX=2
LL=0 or
KK=1
INDX=3
δ



♂
INDX=2
Y
N
PRINTOUT
'KK th Process'
RA(1)=4.0
RA(2)=5.0
RA(3)=6.0
RA(4)=5.0
IX1=IXØ*63342
Attitude Propagation
W
(NEWØ)
WØ into QW
(QTN)
$w_{k-1} \circ s_{k-1} = s_k$
(QXQ)
New attitude
§
§
$w_{k-1} \circ \hat{\hat{s}}_{k-1} = \hat{s}_k$
(QXQ)
Predicted attitud~
QS1 → QS
RR → RW
Observation noise V
and modification of V(4)
(GAUSS4)
Deviation V1 and V2
of AD
(GAUSS4)
$ReD^{l}_{mn}(s_k), ImD^{l}_{mn}(s_k)$
(DRI2)
τ


τ
AD(=a)
(ADC)
Coefficients AA,BB
of harmonies of $h(s_k,k)$
(COEFF1)
$mk(i)=h^i(s_{k},_k)+v_k(i)$
$1\leq i\leq 4$
(FU)
$ReD^1_{mn}(\hat{\hat{s}}_k),ImD^1_{mn}(\hat{s}_k)$
(DRI2)
$m_k(i)=h^k(\hat{s}_k, K)$
$5\leq j\leq 6$
(FU)
σ


α
P1,Q1 from PØ,QØ
(NPAQ)
$\hat{s}_k$, $\hat{\hat{S}}_k$, error 1,
error2, error3, error4
(ERR)
INDX=3
N
Y
ER1 → ERRN
ER2 → ERH1
ER3 → ERH2
ER4 → ER12
s_k → QSN
$\hat{s}_k$ → QH1
$\dot{s}_k$ → QH2
Reparametrization
P1, Q1 PØ, QØ
SH2 SHØ
ε


ε
KK=KK+1
INDX=3
Y
N
MM=MM+1
Data
Into
Disk
(TRANS)
N
KK > NX
Y
ι
K
PRINTOUT
all the result
(TRANS)
STOP

ADC

A Subprogram

Purpose:

To choose two appropriate sets of apparent direction cosines AD with respect to the earth-centered inertial coordinate system.

Calling Sequence:

CALL ADC(AXX, DD, V1, V2, AD)

Common Variables:

None

Input:

AXX	2x3x3-dim. array representing the orientation of the two sets of the start tracker base reference axes with respect to the satellite base reference axes, given in the main program, $A_{x^t x}$.
DD	3x3-dim. array representing $\{ D^1_{mn}(s_k) \}$, computed by 'DRI2'.
V1,V2	4-dim. arrays representing random numbers from the normal distribution.

Output:

AD	2x3-dim. array representing apparent direction cosines, a(k).

Special Considerations:

1. First AD(=a) is chosen to satisfy the relation

$$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = A_{x^t x} \cdot A_{xz^I} \cdot a$$

then a is replaced by a+v and normalized. In the program, the orthogonality of $A_{x^t x}$, A_{xz^I} is used.

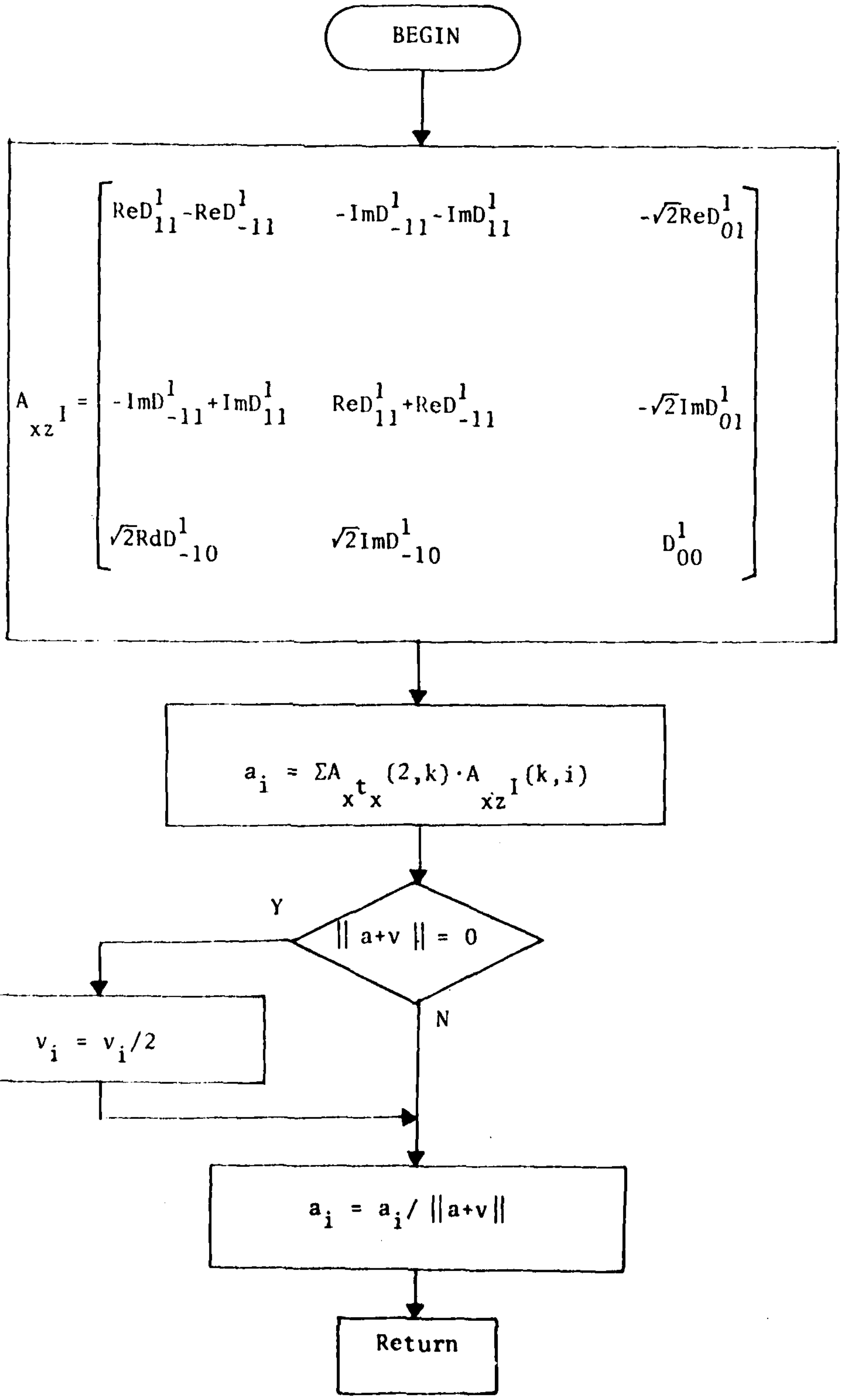
2. v is of normal distribution with a relatively small variance for the realistic simulations.

3. If $\| a + v \| = 0$, then v_i is replaced by $v_i/2$ to avoid $\| a + v \| = 0$.

Other Subprograms Called:

None.

Flow Chart


BEGIN
$A_{xz^I} = \begin{bmatrix} ReD^1_{11}-ReD^1_{-11} & -ImD^1_{-11}-ImD^1_{11} & -\sqrt{2}ReD^1_{01} \\ -ImD^1_{-11}+ImD^1_{11} & ReD^1_{11}+ReD^1_{-11} & -\sqrt{2}ImD^1_{01} \\ \sqrt{2}RdD^1_{-10} & \sqrt{2}ImD^1_{-10} & D^1_{00} \end{bmatrix}$
$a_i = \Sigma A_{x^t x}(2,k) \cdot A_{xz^I}(k,i)$
Y
$\| a+v \| = 0$
N
$v_i = v_i/2$
$a_i = a_i / \|a+v\|$
Return

BLKØ

A Subprogram

Purpose:

To determine a region and parameters for local integration adjusting a block region determined by 'DMNØ'.

Calling Sequence:

```
CALL         BLKØ (A, H, Z, T, W, NZ, NT, NW)
```

Common Variables:

None

Input:

A	3-dim array representing the mode of the conditional density function.
Z, T, W	2-dim arrays representing the coordinates of 8 corner points of a rectangular region.

Output:

Z, T, W	2-dim arrays containing the coordinates of a modified rectangular region.
NZ, NT, NW	integers representing the numbers of meshes in Z, T, W-directions, respectively.

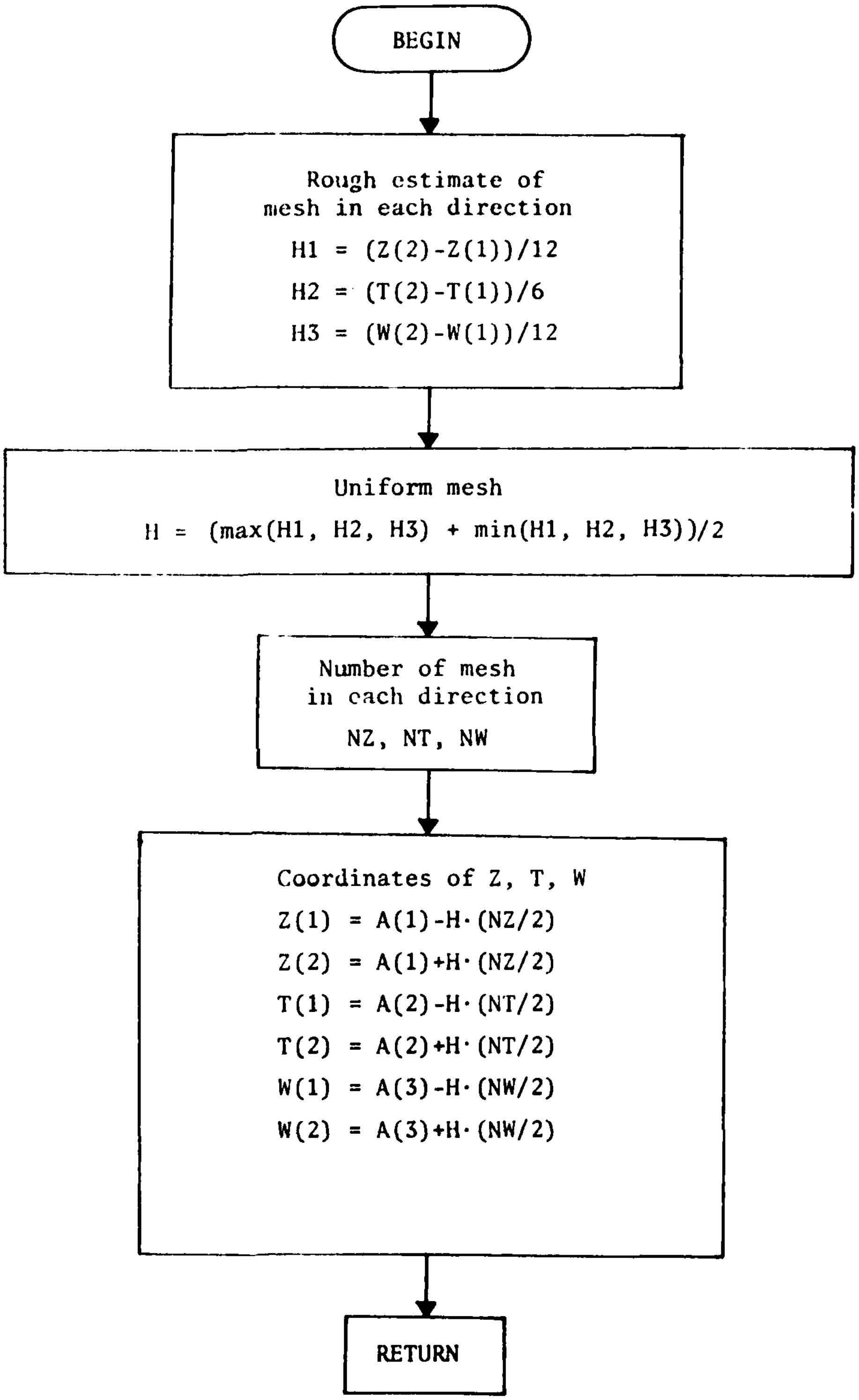
Special Considerations:

1. The numbers of mesh in Z, T, W-directions are adjusted so that they approximately become 12, 6, 12 or 6, 6, 6 according to the distribution.
2. Each number of NZ, NT, NW is a multiple of 6.

Other subprograms called:

None.

Flow Chart


BEGIN
Rough estimate of
mesh in each direction
H1 = (Z(2)-Z(1))/12
H2 = (T(2)-T(1))/6
H3 = (W(2)-W(1))/12
Uniform mesh
H = (max(H1, H2, H3) + min(H1, H2, H3))/2
Number of mesh
in each direction
NZ, NT, NW
Coordinates of Z, T, W
Z(1) = A(1)-H·(NZ/2)
Z(2) = A(1)+H·(NZ/2)
T(1) = A(2)-H·(NT/2)
T(2) = A(2)+H·(NT/2)
W(1) = A(3)-H·(NW/2)
W(2) = A(3)+H·(NW/2)
RETURN

COEFF1

A Subprogram

Purpose:

To set up tables of the coefficients of the following six functions expressed as harmonic series on SO(3),

$$[-u, u_3, -\underline{\underline{u}}_1 \underline{\underline{u}}_3, \underline{u}_2, \underline{\underline{u}}_2]^T$$

where $u_j(k)$ is the j^{th} direction cosine of a star tracker at time k and is given by

$$u_j(k)=m_{j3}\cdot a_3(k)\cdot D^1_{00}(s_k)+[\sqrt{2}\ m_{j3}a_1(k)ReD^1_{-10}(s_k)$$

$$+\ (m_{j2}\cdot a_2(k)-m_{j1}\cdot a_1(1))\ ReD^1_{-11}(s_k)-\ \sqrt{2}m_{j1}a_3(k)ReD^1_{01}(s_k)$$

$$+\ (m_{j1}\cdot a_1(k)+m_{j2}\cdot a_2(k))ReD^1_{11}(s_k)]$$

$$+[\sqrt{2}\ m_{j3}\cdot a_2(k)\ \ ImD^1_{-10}(s_k)-(m_{j1}\cdot a_2(k)\ +\ m_{j2}\cdot a_1(k))ImD^1_{-11}(s_k)$$

$$-\ \sqrt{2}\ m_{j2}\cdot a_3(k)\cdot ImD^1_{01}(s_k)\ +\ (m_{j2}a_1(k)\ -\ m_{j1}\cdot a_2(k))ImD^1_{11}(s_k)$$

Calling Sequence:

```
CALL    COEFF1(AXX, AD, AA, BB)
```

Common Variables:

None

Input:

AXX   Two sets of star tracker base reference axes with respect to the satellite base reference axes, given in the main program, 2x3x3 dimensional array, AXX(1,i,j) for the first start tracker and AXX(2,i,j) for the second, m_{ij} in the equation of $u_j(k)$.

AD — 2x3 dimensional array representing the apparent direction cosines with respect to the earth-centered inertial set of coordinate system, given in the main program, $a_i(k)$ in the equation $u_j(k)$.

Output:

AA — 6x3x3 dimensional array representing the coefficients of $\mathrm{ReD}^1_{mn}(s_k)$ in $-\underline{u}_1$, $\underline{u}_3$, $-\underline{\underline{u}}_1$, $\underline{\underline{u}}_3$, $\underline{u}_2$ and $\underline{\underline{u}}_2$.

BB — 6x3x3 dimensional array representing the coefficients of $\mathrm{ImD}^1_{mn}(s_k)$ in the six functions.

Special Considerations:

1. The coefficients are arranged and stored in AA and BB. For example, the coefficients of $-\underline{u}_1$ are stored in AA $(1, \cdot, \cdot)$ as follows

$$AA(1,\cdot,\cdot) = \begin{bmatrix} -(m_{j1}a_1+m_{j2}a_2) & -\sqrt{2}m_{j3}a_1 & -(m_{j2}a_2 - m_{j1}a_1) \\ -\sqrt{2}m_{j1}a_3 & -2m_{j3}a_3 & \sqrt{2}m_{j1}a_3 \\ -(m_{j2}a_2-m_{j1}a_1) & \sqrt{2}m_{j3}a_1 & -(m_{j1}a_1 + m_{j2}a_2) \end{bmatrix}$$

where the coefficients of $\mathrm{ReD}^1_{mn}(s_k)$ are placed in A(1, m+2, n+2).

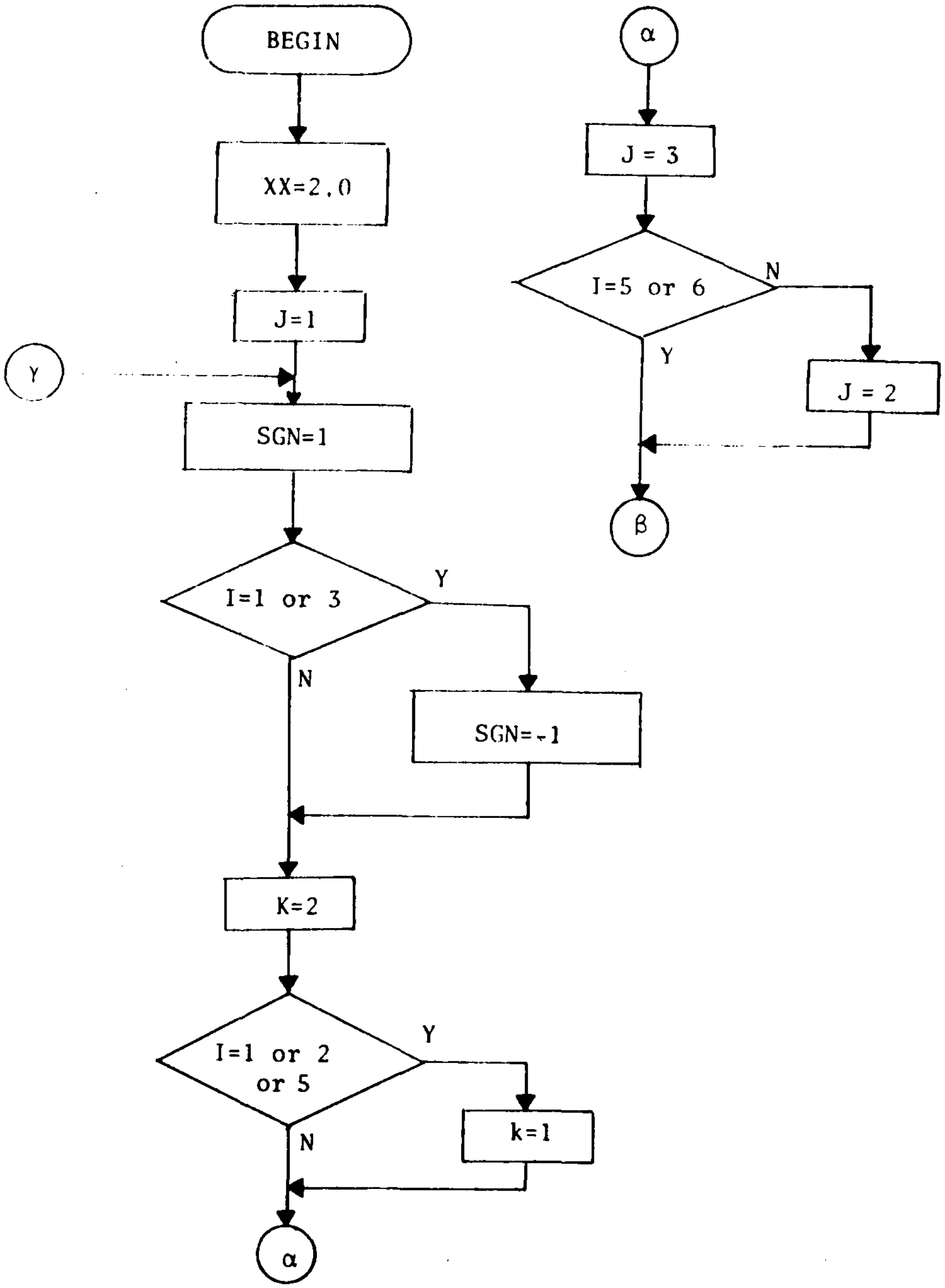
The arrangement of coefficients are made, so that one can express $u_j(k)$ as an expansion of $\{\mathrm{ReD}^1_{mn}(s_k)\}$ and $\{\mathrm{ImD}^1_{mn}(s_k)\}$ for $-1 \le m,n \le 1$, not only under the condition on m, n: $m < n$ or $m=n$ and $m \geq 0$, which is given in the original equation $u_j(k)$. Considering the fact $\mathrm{ReD}^1_{mn}(s_k)=(-1)^{n-m}\mathrm{Red}^1_{-m-n}(s_k)$ and $\mathrm{ImD}^1_{mn}(s_k)=(-1)^{n-m+1}\mathrm{ImD}^1_{-m-n}(s_k)$, we use the following matrix form to see the sign difference between the elements (m+2,n+2) and (-m+2,-n+2).

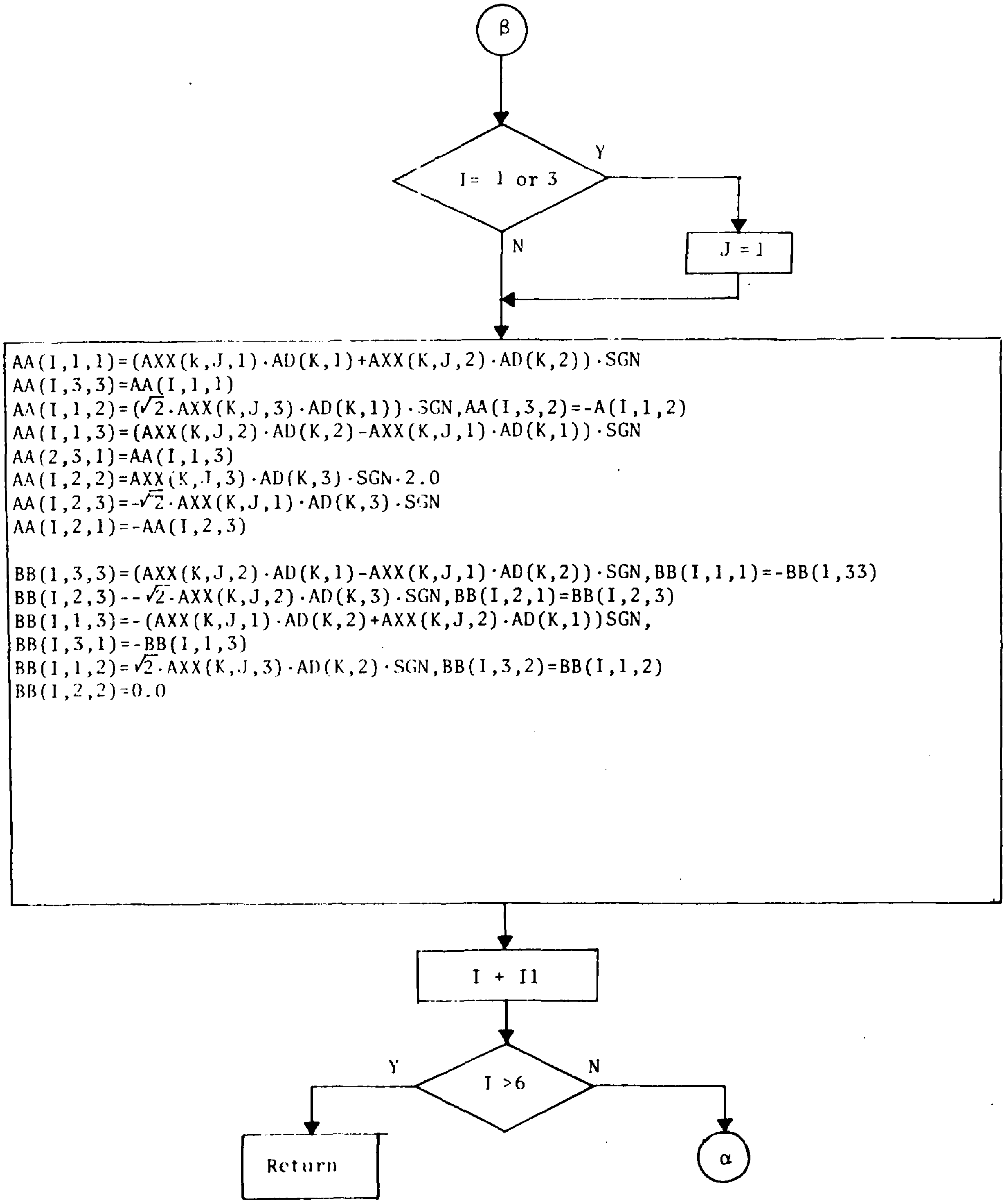
$$\begin{bmatrix} 1 & 1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \end{bmatrix} \text{ for the real part and } \begin{bmatrix} -1 & 1 & 1 \\ 1 & 0 & 1 \\ -1 & 1 & 1 \end{bmatrix}$$

for the imaginary part.

Other Subprograms Called:

None.


BEGIN
XX=2.0
J=1
γ
SGN=1
I=1 or 3
Y
N
SGN=-1
K=2
I=1 or 2
or 5
Y
N
k=1
α
α
J = 3
I=5 or 6
N
Y
J = 2
β



```
AA(I,1,1)=(AXX(k,J,1)·AD(K,1)+AXX(K,J,2)·AD(K,2))·SGN
AA(I,3,3)=AA(I,1,1)
AA(I,1,2)=(√2·AXX(K,J,3)·AD(K,1))·SGN,AA(I,3,2)=-A(I,1,2)
AA(I,1,3)=(AXX(K,J,2)·AD(K,2)-AXX(K,J,1)·AD(K,1))·SGN
AA(2,3,1)=AA(I,1,3)
AA(I,2,2)=AXX(K,J,3)·AD(K,3)·SGN·2.0
AA(I,2,3)=-√2·AXX(K,J,1)·AD(K,3)·SGN
AA(I,2,1)=-AA(I,2,3)

BB(1,3,3)=(AXX(K,J,2)·AD(K,1)-AXX(K,J,1)·AD(K,2))·SGN,BB(I,1,1)=-BB(1,33)
BB(I,2,3)--√2·AXX(K,J,2)·AD(K,3)·SGN,BB(I,2,1)=BB(I,2,3)
BB(I,1,3)=-(AXX(K,J,1)·AD(K,2)+AXX(K,J,2)·AD(K,1))SGN,
BB(I,3,1)=-BB(I,1,3)
BB(I,1,2)=√2·AXX(K,J,3)·AD(K,2)·SGN,BB(I,3,2)=BB(I,1,2)
BB(I,2,2)=0.0
```

C3J

A Subprogram

Purpose:

To compute $\begin{pmatrix} J1 & J2 & J3 \\ M & N & P \end{pmatrix}$

$$= (-1)^{2J1-J2+N} \cdot \frac{(J1+J2-J3)!(j2+J3-J1)!(J3+J1-J2)!(J3+P)!(J3-P)!}{(J1-J2+J3+1)!(J1+m)!(J1-M)!(J2+N)!(J2-N)!}$$

$$\cdot \sum_{t} (-1)^{t} \frac{(J3+J1-N-t)!(J2+N+t)!}{(J3+P-t)!(t+J2-J1-P)!t!(J3-J2+J1-t)!}$$

. where $ML \leq t \leq MU$

$ML = MAX(0, J1+P-J2, -J2-N)$

$MU = MIN\ (J3+J1-N, J3+P, J3-J2+J1)$

Calling Sequence:

Y = C3J(J1, J2, J3, M, N, P)

Input:

J1, J2, J3 non-negative integer with a condition $\max(J1-J2,\ J2-J1) \leq J3 \leq J1+J2$

M, N, P integers with conditions

$-J1 \leq M \leq J1$

$-J2 \leq N \leq J2$

$-J3 \leq P \leq J3$

Common Variables:

None.

Special Considerations:

1. If MU < ML, then Y = 0.0

2. In order to avoid the case

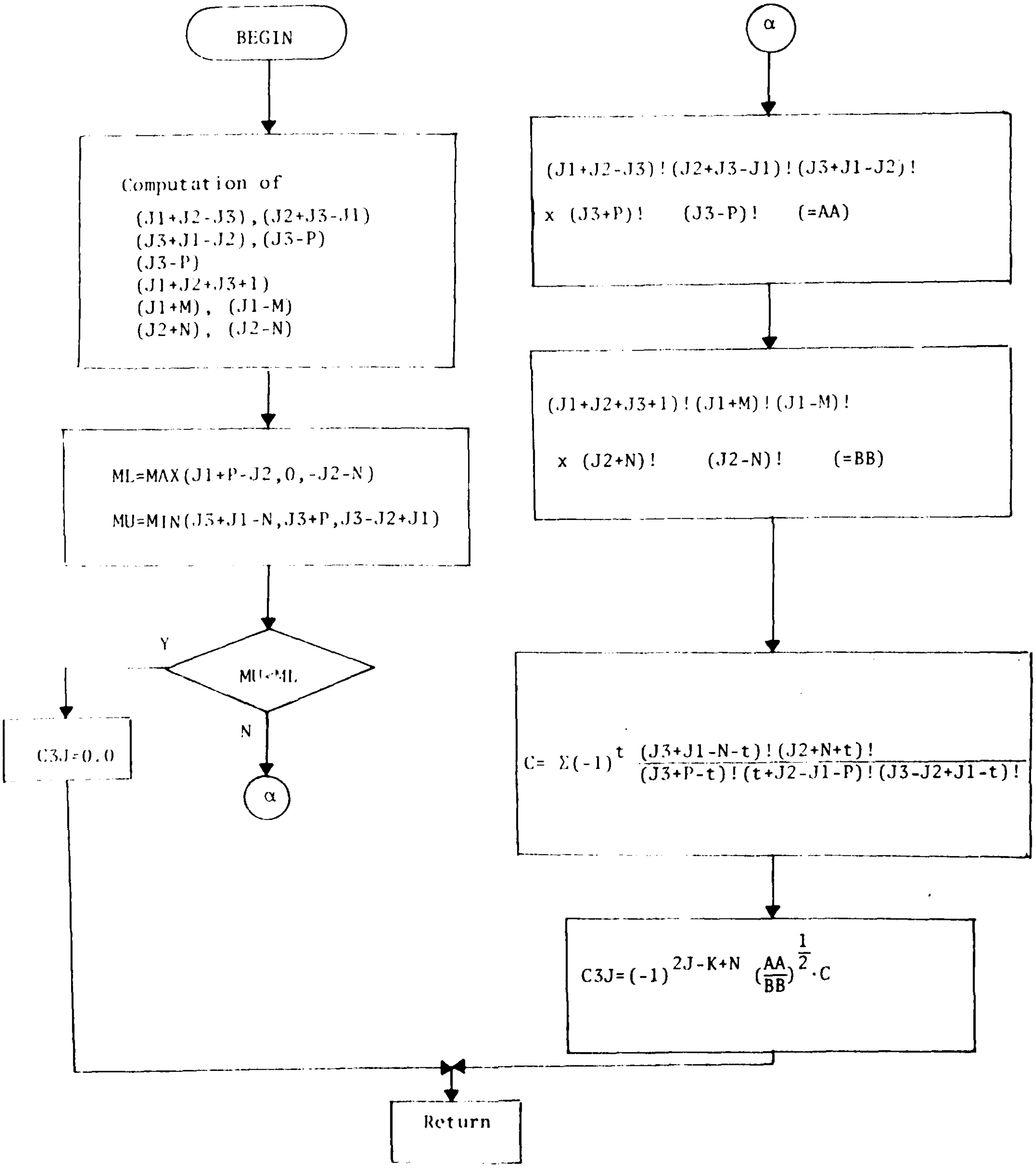
   (-1) * * 0, (for example (-1) * * KJ),

   KJ is replaced by KJ+ a large even integer.

Other Subprograms Called:

FR.

Flow Chart


BEGIN
Computation of
(J1+J2-J3),(J2+J3-J1)
(J3+J1-J2),(J3-P)
(J3-P)
(J1+J2+J3+1)
(J1+M), (J1-M)
(J2+N), (J2-N)
ML=MAX(J1+P-J2,0,-J2-N)
MU=MIN(J3+J1-N,J3+P,J3-J2+J1)
Y
MU<ML
N
C3J=0.0
α
α
(J1+J2-J3)!(J2+J3-J1)!(J3+J1-J2)!
x (J3+P)! (J3-P)! (=AA)
(J1+J2+J3+1)!(J1+M)!(J1-M)!
x (J2+N)! (J2-N)! (=BB)
$C= \Sigma(-1)^t \frac{(J3+J1-N-t)!(J2+N+t)!}{(J3+P-t)!(t+J2-J1-P)!(J3-J2+J1-t)!}$
$C3J=(-1)^{2J-K+N} (\frac{AA}{BB})^{\frac{1}{2}} \cdot C$
Return

DMNØ

A Subprogram

Purpose:

To determine a box region around the mode of the conditional density function $P(\phi, \theta, \psi) = C \cdot \exp(f(\phi, \theta, \psi))$ for local integrations.

Calling Sequence:

CALL DMNØ (P, Q, A, FMAX, Z, T, W)

Common Variables:

PAI2

Input:

PAI2 2π

P, Q $\{P^{\ell k}_{1mn}\}$, $\{P^{\ell k}_{2mn}\}$, 2x5x5-dim arrays, respectively

A Mode $(\phi_o, \theta_o, \psi_o)$ of the function $f(\phi, \theta, \psi)$ obtained in 'MAXI', 3-dim array

FMAX Maximum (f(s)) on SO(3)

Output:

Z First Euler angles ϕ_1, ϕ_2, 2-dim array

T Second Euler angles θ_1, θ_2, 2-dim array

W Third Euler angles ψ_1, ψ_2, 2-dim array
(Z(i), T(i), W(k)) $1 \le i, j, k \le 2$ represents coordinates of a corner of the resulting box region

Special Consideration:

1. Z(1)< A(2), T(1) < T(2), W(1) < W(2)

2. ϕ_1 is determined by decreasing ϕ in $f(\phi, \theta_o, \psi_o)$ until it reaches a constant C_o. On the other hand ϕ_2 is determined by increasing ϕ in $f(\phi, \theta_o, \psi_o)$. The other Euler angles θ_1, θ_2, ψ_1, ψ_2 are obtained similarly.

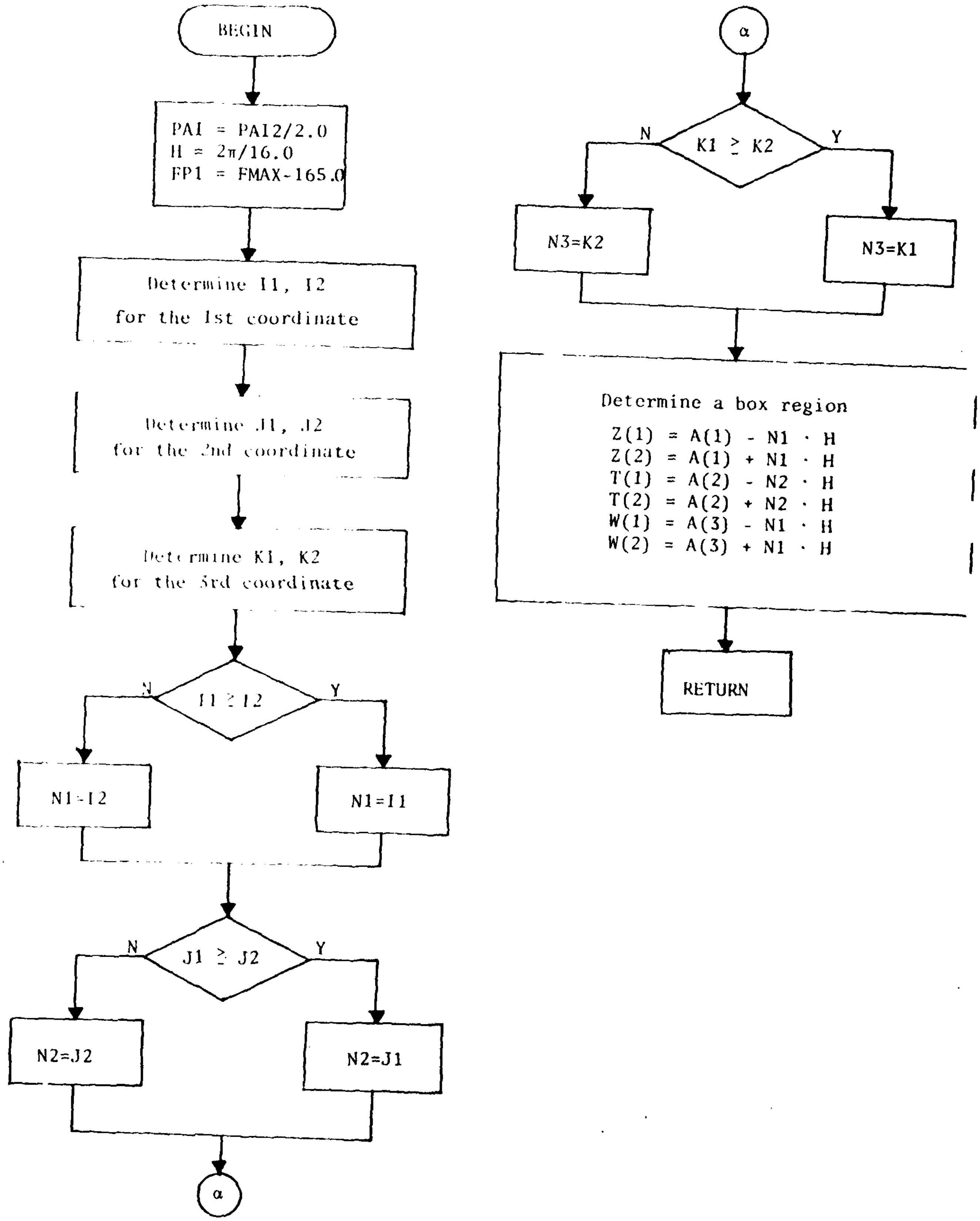
3. The constant C_o is computed by

$$C_o = \begin{cases} \text{Max } f(\phi, \theta, \psi) - 175.0 & \text{if Max } f-165.0 \geq 0 \\ -10.0 & \text{if Max } f-165.0 < 0 \end{cases}$$

Other Subprograms Called:

FP.

Flow Chart


BEGIN
PAI = PAI2/2.0
H = 2π/16.0
FP1 = FMAX-165.0
Determine I1, I2 for the 1st coordinate
Determine J1, J2 for the 2nd coordinate
Determine K1, K2 for the 3rd coordinate
I1 ≥ I2
N
Y
N1=I2
N1=I1
J1 ≥ J2
N
Y
N2=J2
N2=J1
α
α
K1 ≥ K2
N
Y
N3=K2
N3=K1
Determine a box region
Z(1) = A(1) - N1 · H
Z(2) = A(1) + N1 · H
T(1) = A(2) - N2 · H
T(2) = A(2) + N2 · H
W(1) = A(3) - N1 · H
W(2) = A(3) + N1 · H
RETURN

DRI1

A subprogram

Purpose:

To assign values $\mathrm{Re}(D^{\ell}_{mn}(Z,T,W))$ or $\mathrm{Im}(D^{\ell}_{mn}(Z,T,W))$ to ℓ,m,n,Z,T,W

Calling Sequence:

Y = DRI(L, M, N, Z, SD, W, JJ)

Common Variables

PAI2

Input:

PAI2	constant given in the main program
L	order of function $D^{\ell}_{mn}(Z,T,W)$ to be computed
M,N	indices of function $D^{\ell}_{mn}(Z,T,W)$ to be computed
SD	value $d^{\ell}_{mn}(T)$
Z,W	Euler angles with $0 \leq Z,W \leq 2\pi$

JJ	indicator
	if JJ=1, $\mathrm{Re}(D^{\ell}_{mn}(Z,T,W))$ to be computed
	if JJ=2, $\mathrm{Im}(D^{\ell}_{mn}(Z,T,W))$ to be computed

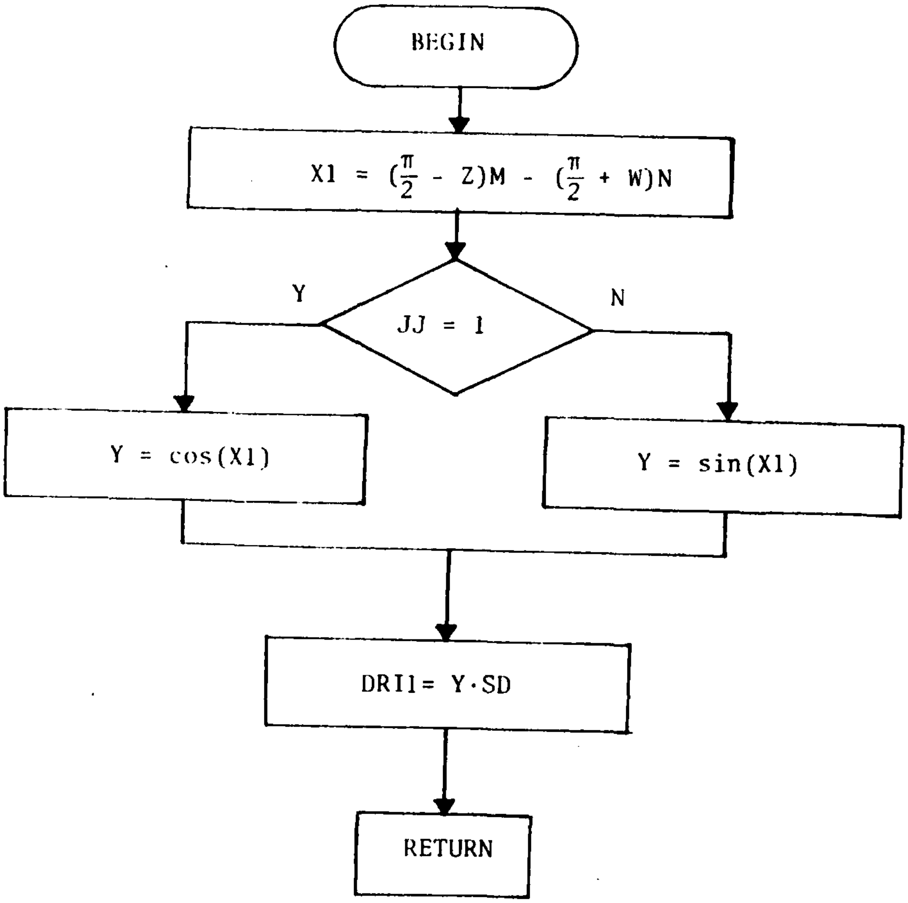
Special Considerations:

1. SD must be precomputed in the calling program. This is taken in order to save computing time in integration.

Other Subprograms Called:

None

F.7. Flow Chart



DRI2

A Subprogram

Purpose:

To calculate $\mathrm{ReD}^1_{mn}(s)$ and $\mathrm{ImD}^1_{mn}(s)$ at $s=(q_1,q_2,q_3,q_4)$ and store the results in the matrix DD. The following relations are used.

$$\left.\begin{aligned}
q_1^2 &= \tfrac{1}{4} - D^1_{00}/4.0 - \mathrm{ReD}^1_{-11}/2.0 \\
q_2^2 &= \tfrac{1}{4} - D^1_{00}/4 + \mathrm{ReD}^1_{-11}/2.0 \\
q_3^2 &= \tfrac{1}{4} + D^1_{00}/4 - \mathrm{ReD}^1_{11}/2.0 \\
q_4^2 &= \tfrac{1}{4} + D^1_{00}/4 + \mathrm{ReD}^1_{11}/2.0 \\
q_1q_2 &= -\mathrm{ImD}^1_{-11}/2 \\
q_1q_3 &= \tfrac{\sqrt{2}}{4}(\mathrm{ReD}^1_{-10}-\mathrm{ReD}^1_{01}) \\
q_2q_4 &= \tfrac{\sqrt{2}}{4}(\mathrm{ReD}^1_{-10}+\mathrm{ReD}^1_{01}) \\
q_3q_4 &= -\mathrm{ImD}^1_{11}/2.0 \\
q_1q_4 &= -\tfrac{\sqrt{2}}{4}(\mathrm{ImD}^1_{-10}+\mathrm{ImD}^1_{01}) \\
q_2q_3 &= -\tfrac{\sqrt{2}}{4}(\mathrm{ImD}^1_{01}-\mathrm{ImD}^1_{-10})
\end{aligned}\right\}
\quad
\begin{aligned}
\mathrm{ReD}^1_{00} &= 1 - 2(q_1^2+q_2^2) \\
\mathrm{ReD}^1_{-11} &= q_2^2 - q_1^2 \\
\mathrm{ReD}^1_{11} &= q_4^3 - q_3^2 \\
\mathrm{ReD}^1_{-10} &= \sqrt{2}(q_1q_3+q_2q_4) \\
\mathrm{ReD}^1_{01} &= \sqrt{2}(q_2q_4-q_1q_3) \\
\mathrm{ImD}^1_{-11} &= -2q_1q_2 \\
\mathrm{ImD}^1_{11} &= -2q_3q_4 \\
\mathrm{ImD}^1_{01} &= -\sqrt{2}(q_1q_4+q_2q_3) \\
\mathrm{ImD}^1_{-10} &= \sqrt{2}(q_2q_3-q_1q_4)
\end{aligned}$$

Calling Sequence:

```
CALL DRI2(Q, DD)
```

Common Variables:

None.

Input:

Q	4-dimensional array representing a quaternion.

Output:

DD 3×3 dimensional array for $\mathrm{ReD}^{\ell}_{mn}(s)$, $\mathrm{ImD}^{\ell}_{mn}(s)$

Special Considerations:

1. The values ReD^{ℓ}_{mn}, ImD_{mn} are stored in the following way

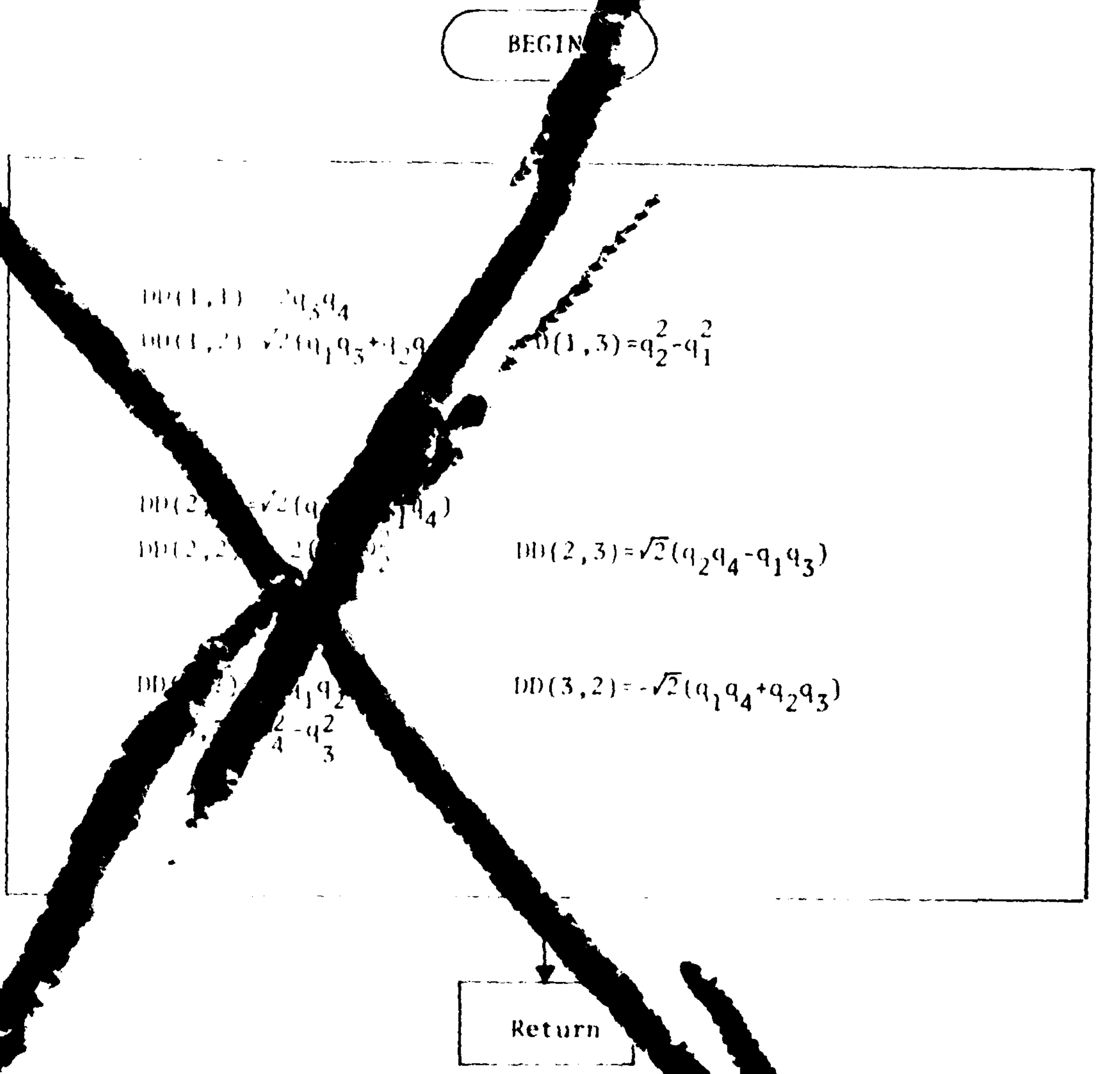
$$\mathrm{DD}(\cdot,\ \cdot) = \begin{bmatrix} \mathrm{ImD}^{1}_{11} & \mathrm{ReD}^{1}_{-10} & \mathrm{ReD}^{1}_{-11} \\ \mathrm{ImD}^{1}_{-10} & \mathrm{ReD}^{1}_{00} & \mathrm{ReD}^{1}_{01} \\ \mathrm{ImD}^{1}_{-11} & \mathrm{ImD}^{1}_{01} & \mathrm{ReD}^{1}_{11} \end{bmatrix}$$

$$= \begin{bmatrix} -2q_3q_4 & \sqrt{2}(q_1q_3+q_2q_4) & q_2^2 - q_1^2 \\ \sqrt{2}(q_2q_3-q_1q_4) & 1-2(q_1^2+q_2^2) & \sqrt{2}(q_2q_4-q_1q_3) \\ -2q_1q_2 & -\sqrt{2}(q_1q_4+q_2q_3) & q_4^2 - q_3^2 \end{bmatrix}$$

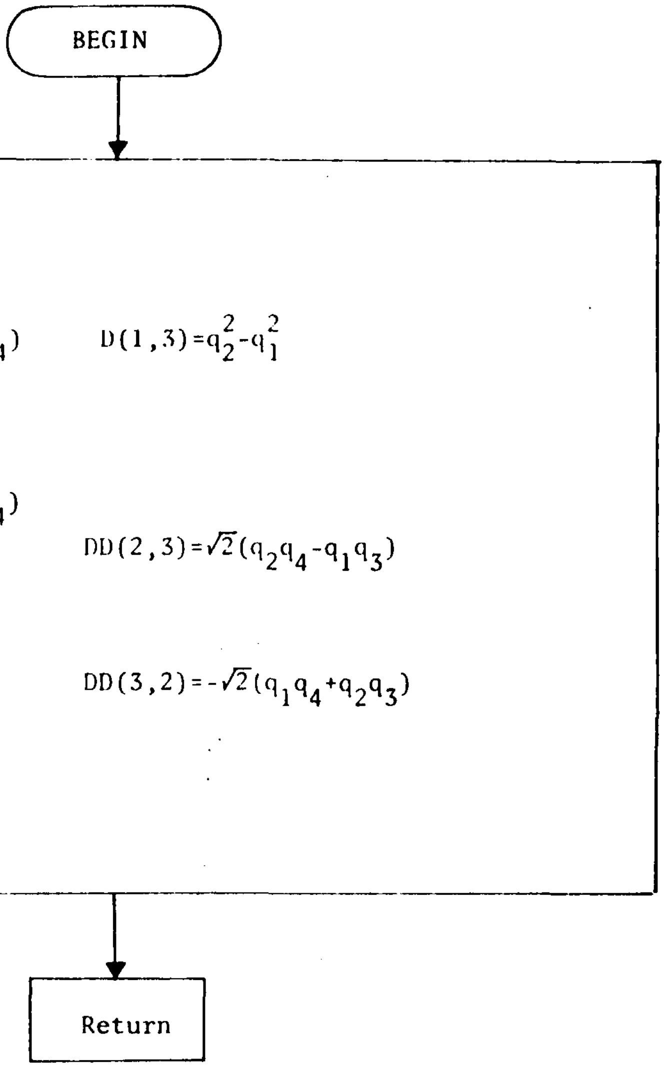
Other Subprograms Called:

None.

Flow Chart


BEGIN
DD(1,1) 2q3q4
DD(1,2) √2(q1q3+q2q
D(1,3)=q2²-q1²
DD(2,
DD(2,2
DD(2,3)=√2(q2q4-q1q3)
DD(3,2)=-√2(q1q4+q2q3)
Return

Flow Chart


BEGIN
$DD(1,1)=-2q_3q_4$
$DD(1,2)=\sqrt{2}(q_1q_3+q_2q_4)$
$D(1,3)=q_2^2-q_1^2$
$DD(2,1)=\sqrt{2}(q_2q_3-q_1q_4)$
$DD(2,2)=1-2(q_1^2+q_2^2$
$DD(2,3)=\sqrt{2}(q_2q_4-q_1q_3)$
$DD(3,1)=-2q_1q_2$
$DD(3,2)=-\sqrt{2}(q_1q_4+q_2q_3)$
$DD(3,3)=q_4^2-q_3^2$
Return

EIGEN

A Subprogram

Purpose:

To compute eigenvaluesand eigenvectors of a given real symmetric matrix by tridiagonalization.

Calling Sequence:

```
CALL EIGEN(N, A, E, V)
```

Common Variables:

None.

Input:

N	dimension of matrix a ($\leq$ 4)
A	2-dim. array containing the symmetric matrix.

Output:

E	array containing the computed eigenvalues in absolute descending order.
V	eigenvectors stored in columns.

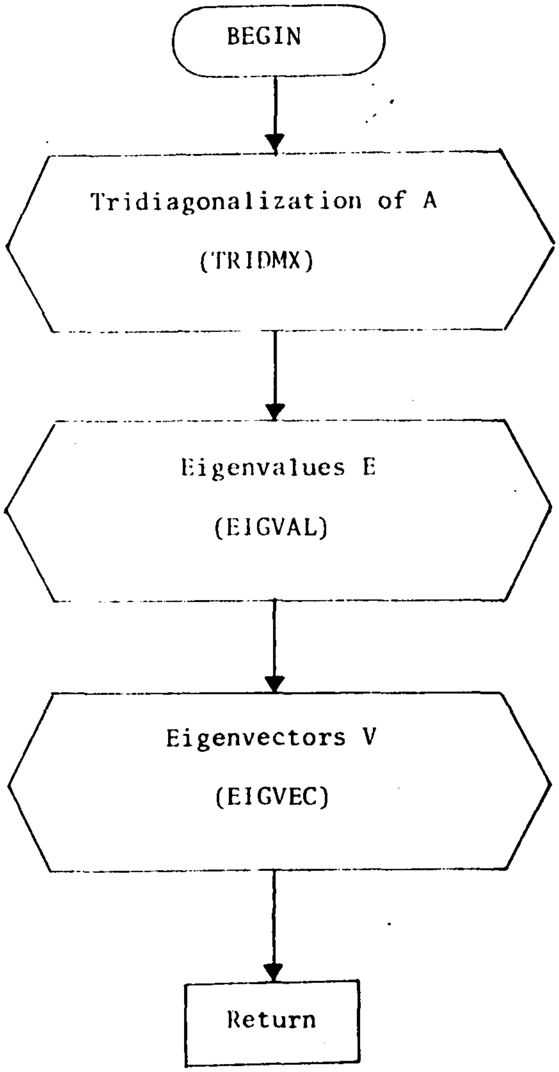
Special Considerations:

None.

Other Subprograms Called:

TRIDMX, EIGVAL, EIGVEC.

Flow Chart


BEGIN
Tridiagonalization of A
(TRIDMX)
Eigenvalues E
(EIGVAL)
Eigenvectors V
(EIGVEC)
Return

EIGVAL

A subprogram

Purpose:

To compute the eigenvalues of a symmetric tridiagonal matrix using Sturn sequences.

Calling Sequence:

```
CALL EIGVAL(LP, E, A, B, W, F)
```

Common Variables

None.

Input:

LP order of tridiagonal matrix

A LP-dim. array containing the diagonal elements of the tridiagonal matrix.

B LP-dim. array containing off-diagonal elements in B(2) through B(N). B(1) = 0.0.

Output:

E LP-dim. array containing the computed eigenvalues in absolute descending order.

W LP-dim. dummy array.

F LP-dim. dummy array.

Special Considerations

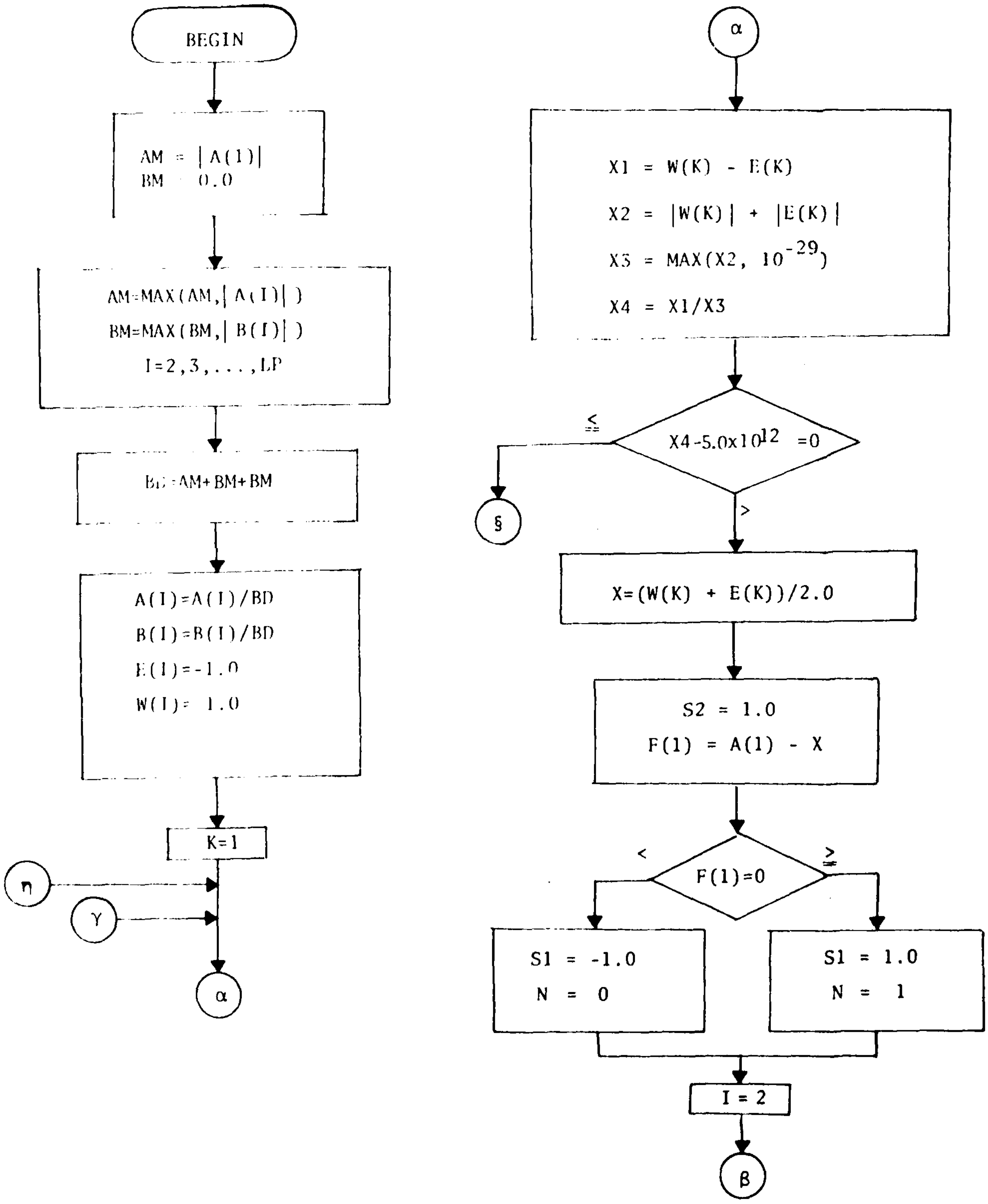
1. The diagonal matrix is not fed into the subroutine as a matrix. Instead, the main diagonal is stored in the A array. The off-diagonal elements are stored in B(2) through B(N). B(1) must be equal to zero.

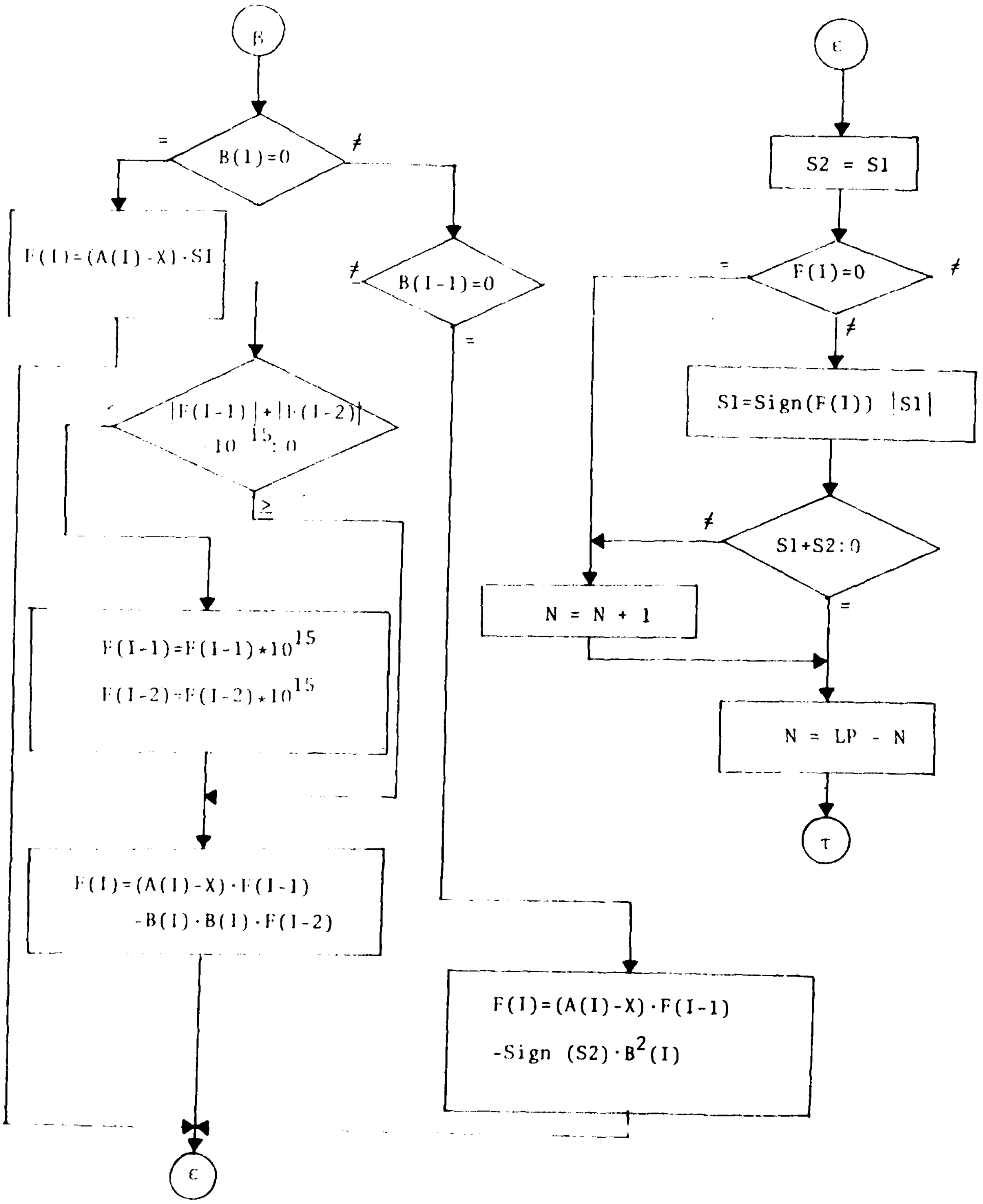
2, EIGVAL is designed to be used after calling TRIDMX. The output from TRIDMX is in the correct form for input to EIGVAL.

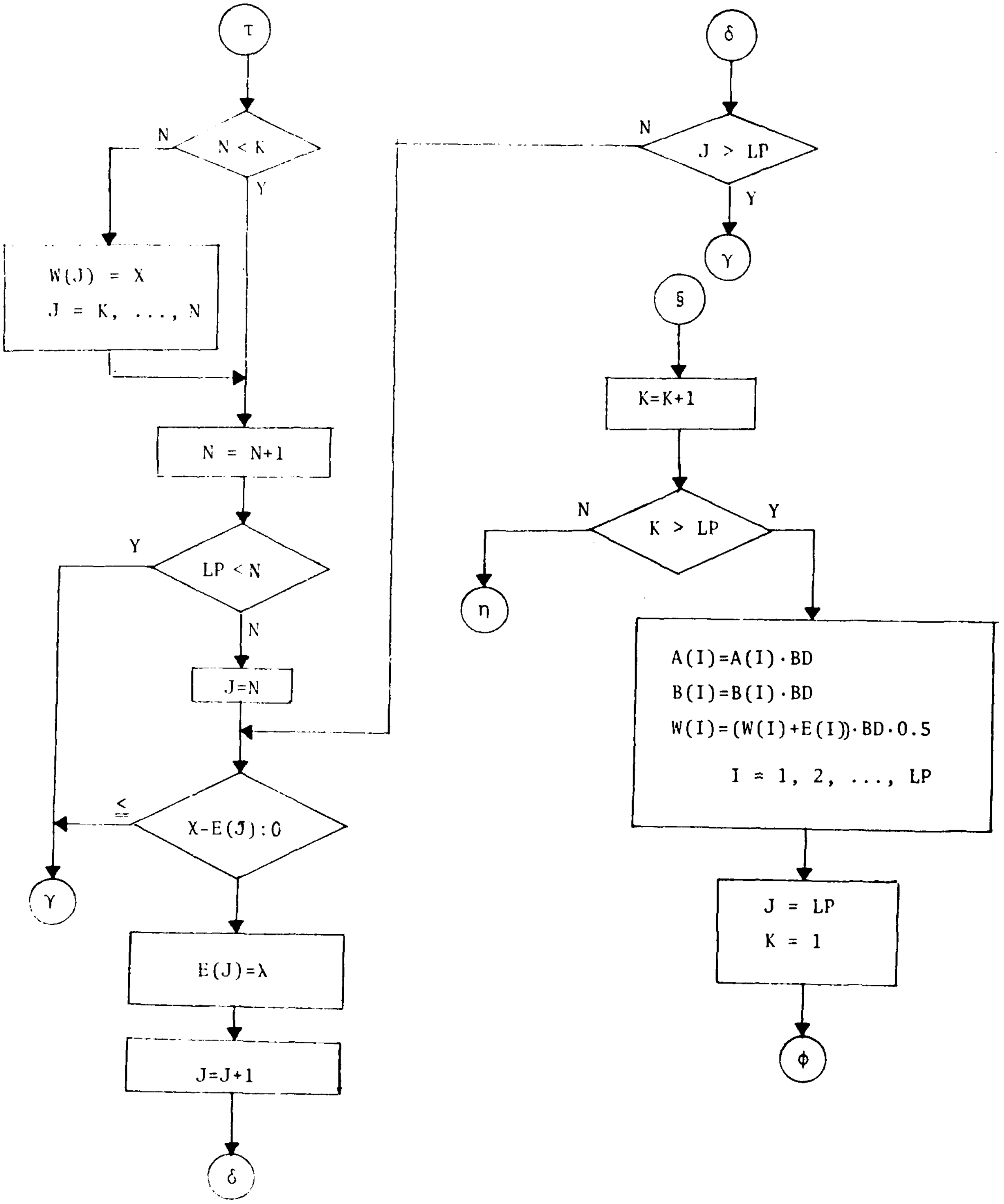
Other Subprograms Called:

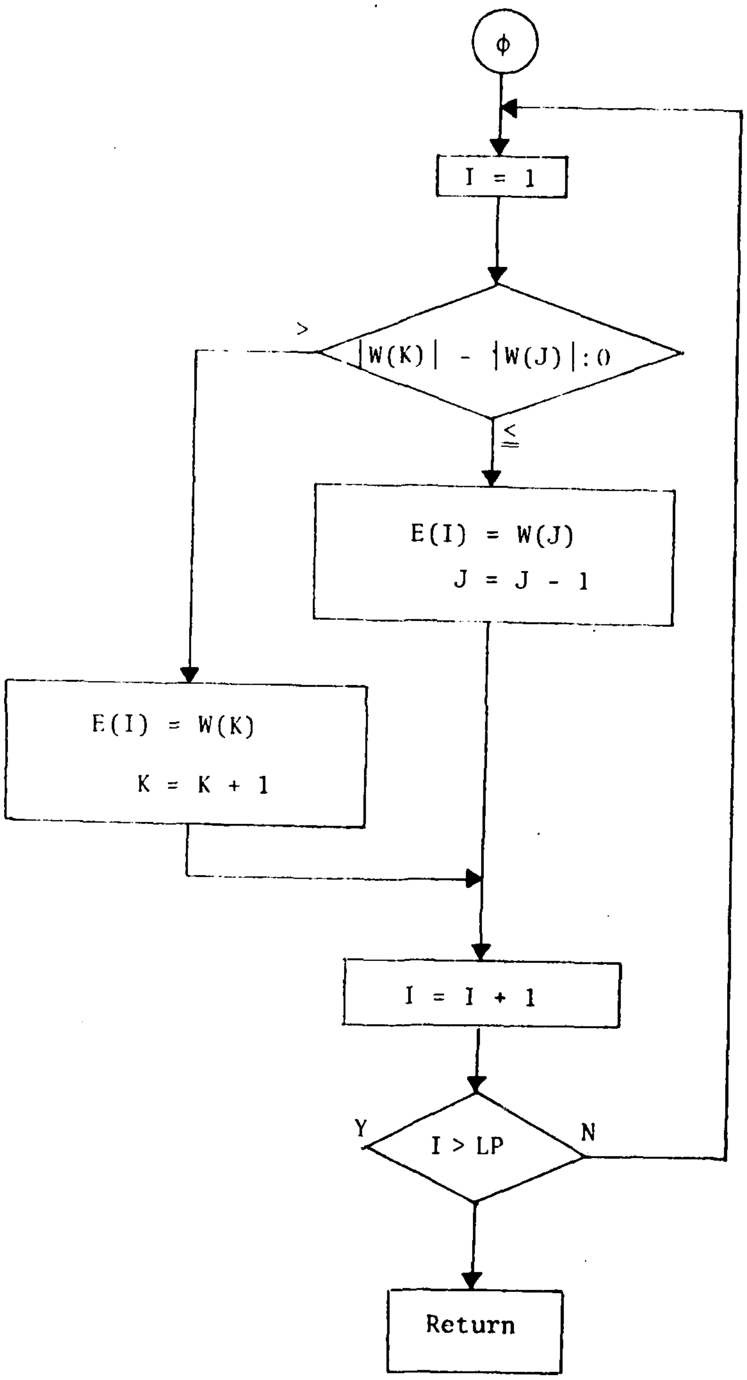
None.

Flow Chart


BEGIN
AM = |A(1)|
BM = 0.0
AM=MAX(AM,|A(I)|)
BM=MAX(BM,|B(I)|)
I=2,3,...,LP
BD=AM+BM+BM
A(I)=A(I)/BD
B(I)=B(I)/BD
E(I)=-1.0
W(I)= 1.0
K=1
η
γ
α
α
X1 = W(K) - E(K)
X2 = |W(K)| + |E(K)|
X3 = MAX(X2, 10^-29)
X4 = X1/X3
≦
X4-5.0x10^12 =0
§
>
X=(W(K) + E(K))/2.0
S2 = 1.0
F(1) = A(1) - X
<
≧
F(1)=0
S1 = -1.0
N = 0
S1 = 1.0
N = 1
I = 2
β


β
B(1)=0
=
≠
F(1)=(A(1)-X)·S1
≠
B(I-1)=0
=
|F(I-1)|+|F(I-2)|
-10^15 : 0
<
≥
F(I-1)=F(I-1)*10^15
F(I-2)=F(I-2)*10^15
F(I)=(A(I)-X)·F(I-1)
-B(I)·B(I)·F(I-2)
F(I)=(A(I)-X)·F(I-1)
-Sign (S2)·B^2(I)
ε
ε
S2 = S1
F(I)=0
=
≠
≠
S1=Sign(F(I)) |S1|
S1+S2:0
≠
=
N = N + 1
N = LP - N
τ


τ
N < K
N
Y
W(J) = X
J = K, ..., N
N = N+1
LP < N
Y
N
J=N
X-E(J):0
≤
γ
E(J)=λ
J=J+1
δ
δ
J > LP
N
Y
γ
§
K=K+1
K > LP
N
Y
η
A(I)=A(I)·BD
B(I)=B(I)·BD
W(I)=(W(I)+E(I))·BD·0.5
I = 1, 2, ..., LP
J = LP
K = 1
φ


ϕ
I = 1
>
|W(K)| - |W(J)|:0
≤
E(I) = W(J)
J = J - 1
E(I) = W(K)
K = K + 1
I = I + 1
Y
I > LP
N
Return

EIGVEC

A subprogram

Purpose:

To compute the eigenvectors of a real symmetric tridiagonal matrix using Wilkinson's method.

Calling Sequence:

```
CALL EIGVEC(LP, NM, R, A, B, E, V, P, Q)
```

Input:

LP order of tridiagonal matrix.

NM maximum number of rows that the tridiagonal matrix can have as specified by the DIMENSION statement in the calling program.

R NMXNM-dim. array containing the transformation vectors used to reduce the symmetric matrix to tridiagonal form.

A N-dim. array containing the diagonal elements of the tridiagonal matrix.

B N-dim. array containing the off-diagonal matrix in B(2) through B(N).

E N-dim. array containing the eigenvalues of the tridiagonal matrix.

Output:

V NMXNM-dim. array containing the eigenvectors stored in columns of the tridiagonal matrix.

P N-dim. dummy array for temporary storage.

Q N-dim. dummy array for temporary storage.

Special Considerations:

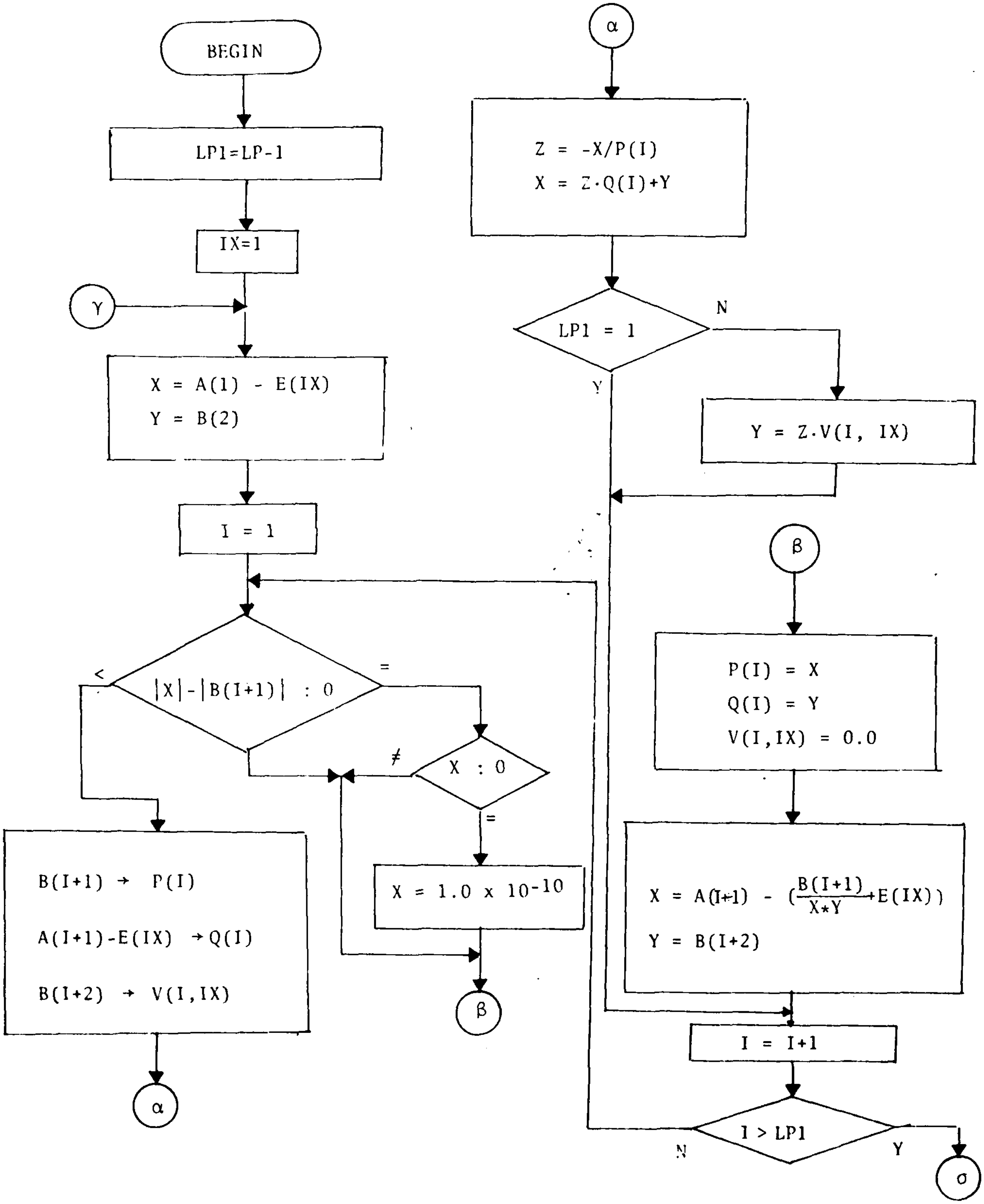
1. EIGVEC is designed to be called after TRIDMX and EIGVAL. The transformation vectors needed in A are computed by TRIDMX. The tridiagonal matrix itself is not fed into the subroutine as a matrix. Rather, the diagonal elements are in the D array and the off-diagonal elements are in the B array. TRIDMX will put those respective elements in those arrays.

2. The accuracy of the eigenvectors is determined by the separation of the eigenvalues. The closer the eigenvalues, the less accurate the eigenvectors. In case of multiple eigenvalues, multiple eigenvectors will be computed.

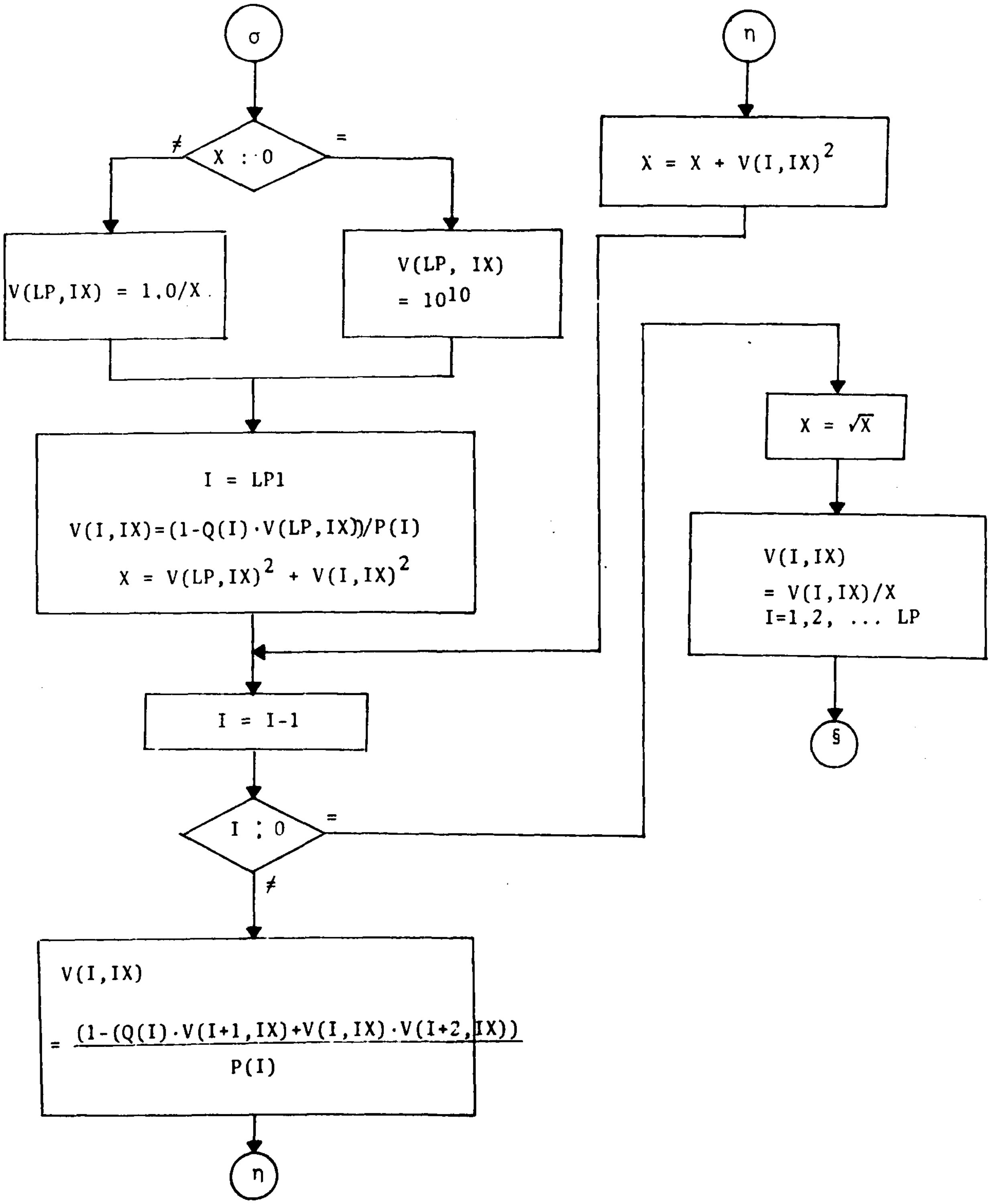
Other Subprograms Called.

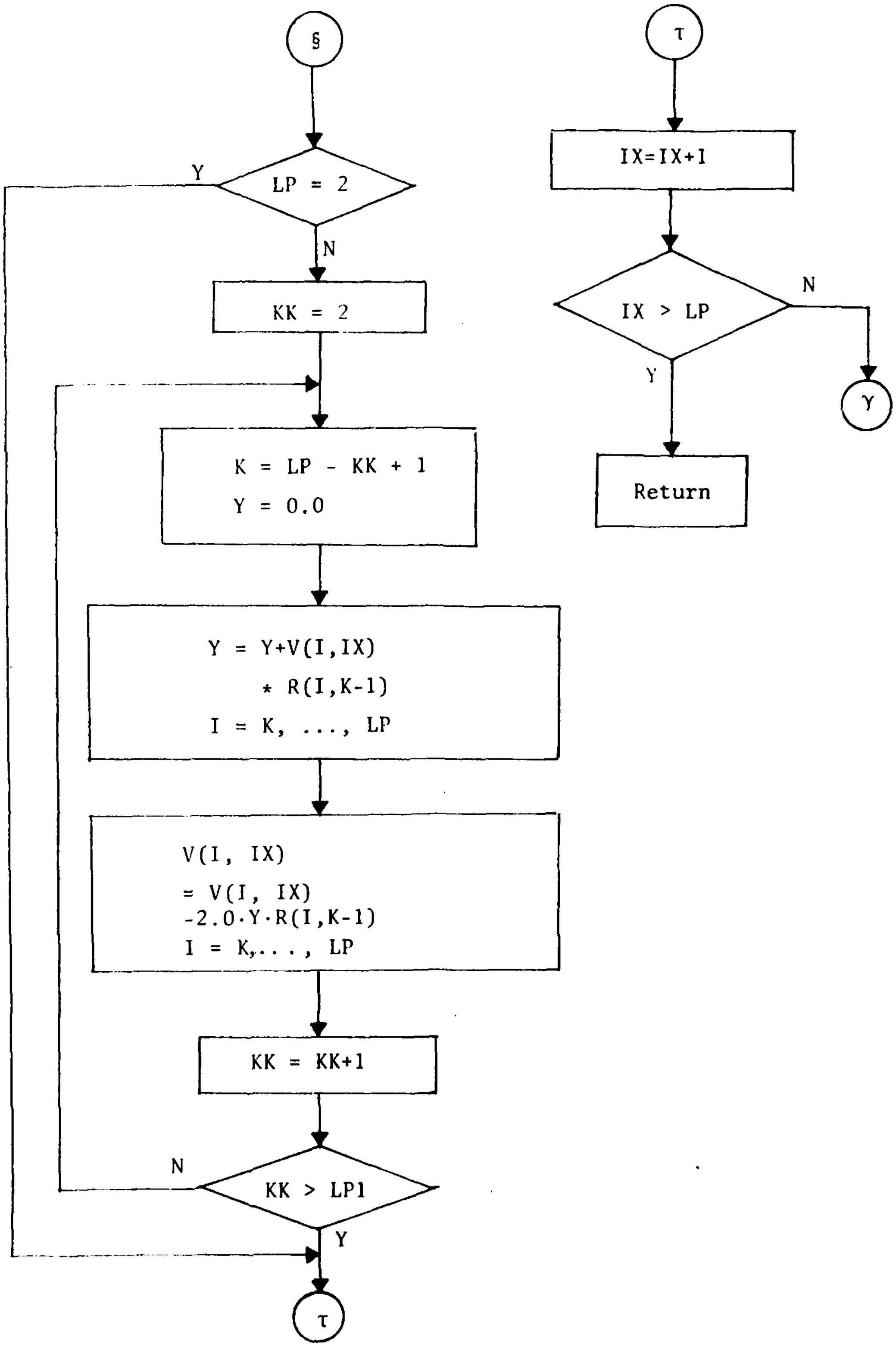
None.

Flow Chart


BEGIN
LP1=LP-1
IX=1
γ
X = A(1) - E(IX)
Y = B(2)
I = 1
<
=
|X|-|B(I+1)| : 0
≠
X : 0
=
X = 1.0 x 10-10
β
B(I+1) → P(I)
A(I+1)-E(IX) →Q(I)
B(I+2) → V(I,IX)
α
α
Z = -X/P(I)
X = Z·Q(I)+Y
LP1 = 1
N
Y
Y = Z·V(I, IX)
β
P(I) = X
Q(I) = Y
V(I,IX) = 0.0
X = A(I+1) - (B(I+1)/X*Y + E(IX))
Y = B(I+2)
I = I+1
I > LP1
N
Y
σ

Flow Chart


σ
X : 0
≠
=
V(LP,IX) = 1.0/X
V(LP, IX) = 10¹⁰
I = LP1
V(I,IX)=(1-Q(I)·V(LP,IX))/P(I)
X = V(LP,IX)² + V(I,IX)²
I = I-1
I : 0
=
≠
V(I,IX) = (1-(Q(I)·V(I+1,IX)+V(I,IX)·V(I+2,IX))/P(I)
η
η
X = X + V(I,IX)²
X = √X
V(I,IX) = V(I,IX)/X
I=1,2, ... LP
§


§
LP = 2
Y
N
KK = 2
K = LP - KK + 1
Y = 0.0
Y = Y+V(I,IX)
* R(I,K-1)
I = K, ..., LP
V(I, IX)
= V(I, IX)
-2.0·Y·R(I,K-1)
I = K,..., LP
KK = KK+1
N
KK > LP1
Y
τ
τ
IX=IX+1
IX > LP
N
Y
Y
Return

ERR

A Subprogram

Purpose:

1. To determine the maximum value of a function in the exponent of the conditional density function $p(s_k|m^k)$ and Euler angles where the function is maximum.

2. To compute the normalizing constant for $p(s_k|m^k)$.

3. To evaluate $E(q \cdot q^T)$ with respect to $p(s_k|m^k)$ and to find the maximum eigenvalue λ of $E(q \cdot q^T)$ and its eigenvector P so that

$$\text{ERR1} = 1 - \lambda = 1 - p^T E(q \cdot q^T) p$$

is the optimal estimation error.

4. To find distances between s_k and $\hat{s}_k$, between s_k and $\hat{\hat{s}}_k$, and between $\hat{s}_k$ and $\hat{\hat{s}}_k$ on SO(3), where $\hat{s}_k$ the optimal estimate and $\hat{\hat{s}}_k$ the mode of the $p(s_k|m^k)$.

Calling Sequence:

```
CALL     ERR(INDX, SDD1, SDD2, P, Q, QH1, QH2, ERR1, ERH1, ERH2,
         ER12)
```

Common Variables:

N, NZ, MT   integers given in the main program.

Input:

INDX   INDX=1   If return without any computation
INDX=2   If the mode of distribution in terms of Euler angles and quaternion $\hat{\hat{s}}_k$ to be computed with error between s_k and $\hat{\hat{s}}_k$.
INDX=3   If the maximum eigenvalue λ of $E(q \cdot q^T)$ and its eigenvector $\hat{s}_k$ with the error estimate between s_k and $\hat{s}_k$ in addition to the case INDX=2.

SDD1 3x3x17 dimensional array, $\{d^1_{mn}(\theta)\}$

SDD2 5x5x17 dimensional array $\{d^2_{mn}(\theta)\}$
both are precomputed in 'INSDD'.

SINE 33-dim. array, $\{\sin(\phi)\}$.

COSN 33-dim. array, $\{\cos(\phi)\}$
both are computed in 'SINCOS'.

P,Q 2x5x5 dimensional arrays representing the Fourier coefficients in $p(s_k|m^k)$, computed in 'NPAQ'.

Output:

QH1 4-dim. array, the unit eigenvector corresponding to the maximum eigenvalue λ , $\hat{s}_k$.

QH2 4-dim. array, the mode of the distribution.

ERR1 $1-\lambda$, where λ is the maximum eigenvalue of $E(q\cdot q^T)$.

ERH1 distance between $s_k = (q_1, q_2, q_3, q_4)$ and $\hat{s}_k = (p_1, p_2, p_3, p_4)$ i.e.

$$1 - \sum_{i=1}^{4} p_i q_i .$$

ERH2 distance between s_k and $\hat{\hat{s}}_k$.

ER12 distance between s_k, $\hat{\hat{s}}_k$.

Special Consideration

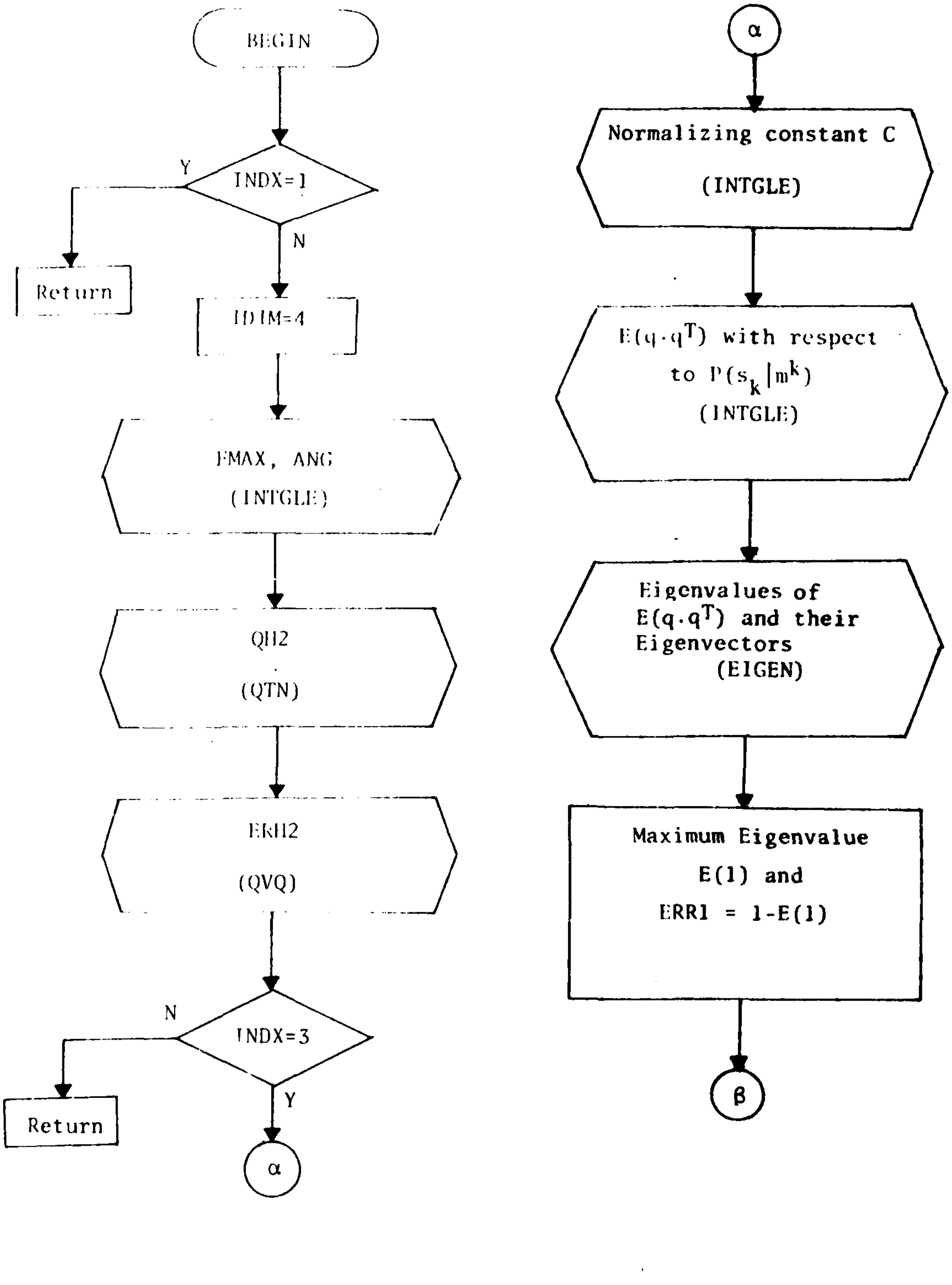
1. $\{d^1_{mn}(\theta)\}$ and $\{\sin(\phi), \cos(\phi)\}$ are precomputed to save computing time.

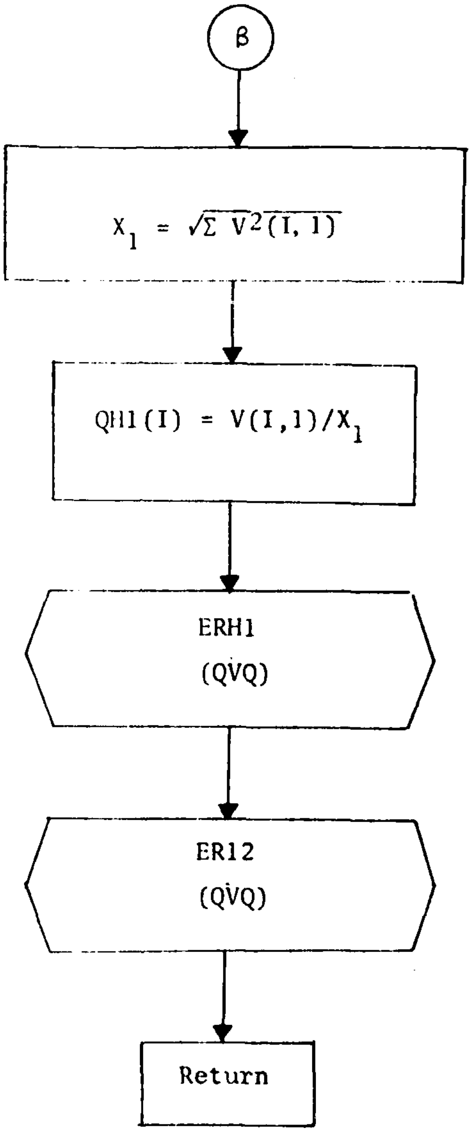
2. To find eigenvalues and eigenvectors of symmetric matrix $E(q\cdot q^T)$, first the matrix is tridiagonalized by 'TRIDMX', then eigenvalues and eigenvectors are obtained by 'EIGVAL', 'EIGVEC' in the subprogram EIGEN.

Other Subprograms Called

INTGLE, QTN, QVQ, EIGEN

Flow Chart


BEGIN
INDX=1
Y
N
Return
IDIM=4
EMAX, ANG
(INTGLE)
QH2
(QTN)
ERH2
(QVQ)
INDX=3
N
Y
Return
α
α
Normalizing constant C
(INTGLE)
$E(q \cdot q^T)$ with respect to $P(s_k|m^k)$
(INTGLE)
Eigenvalues of $E(q \cdot q^T)$ and their Eigenvectors
(EIGEN)
Maximum Eigenvalue E(1) and ERR1 = 1-E(1)
β


β
$X_1 = \sqrt{\Sigma V^2(I, 1)}$
$QH1(I) = V(I,1)/X_1$
ERH1
$(Q\dot{V}Q)$
ER12
$(Q\dot{V}Q)$
Return

ESTIM

A Subprogram

Purpose:

1. To compute the mode and the maximum value of the function $f(\phi,\ \theta,\ \psi)$ in the conditional density function $P(s^k|m^k)$
2. To compute the normalizing constant for $P(s_k|m^k)$by local integration.
3. To evaluate the maximum eigenvalue λ and its eigenvector $p(= \hat{s}_k)$ of the matrix $E(q \cdot q^T)$ with the conditional density function $P(S_k|m^k)$ and obtain the optimal estimation error

   $$1 - \lambda = 1 - p^T \cdot E(q \cdot q^T) \cdot p$$
4. To obtain distances $\rho(s_k,\ \hat{s}_k)$, $\rho(s_k,\ \hat{\hat{s}}_k)$, $\rho(\hat{s}_k,\ \hat{\hat{s}}_k)$, where s_k true quaternions, $\hat{s}_k$ optimal estimates, $\hat{\hat{s}}_k$ maximum likelihood estimates by $P(s_k|m^k)$.

Calling Sequence:

CALL	ESTIM (P, Q, QT, QH1, QH2, ERR1, ERH1, ERH2, ER12)

Common Variables:

QS

Input:

QS	s_k, real attitude, 4-dim array
P, Q	$\{P^{\ell k}_{1mn}\}$, $\{P^{\ell k}_{2mn}\}$, 2x5x5-dim arrays
QT	predicted attitude at time k, 4-dim array

Output:

QH1	Optimal estimate of s_k, $\hat{s}_k$, 4-dim array
QH2	Mode of the distribution, $\hat{\hat{s}}_k$, 4-dim array

ERR1 Optimal estimation error given by $1-\lambda$, where λ is the maximum eigenvalue of $E(q\cdot q^T)$

ERH1 Distance between s_k and $\hat{s}_k$

ERH2 Distance between s_k and $\hat{\hat{s}}_k$

ER12 Distance between $\hat{s}_k$ and $\hat{\hat{s}}_k$.

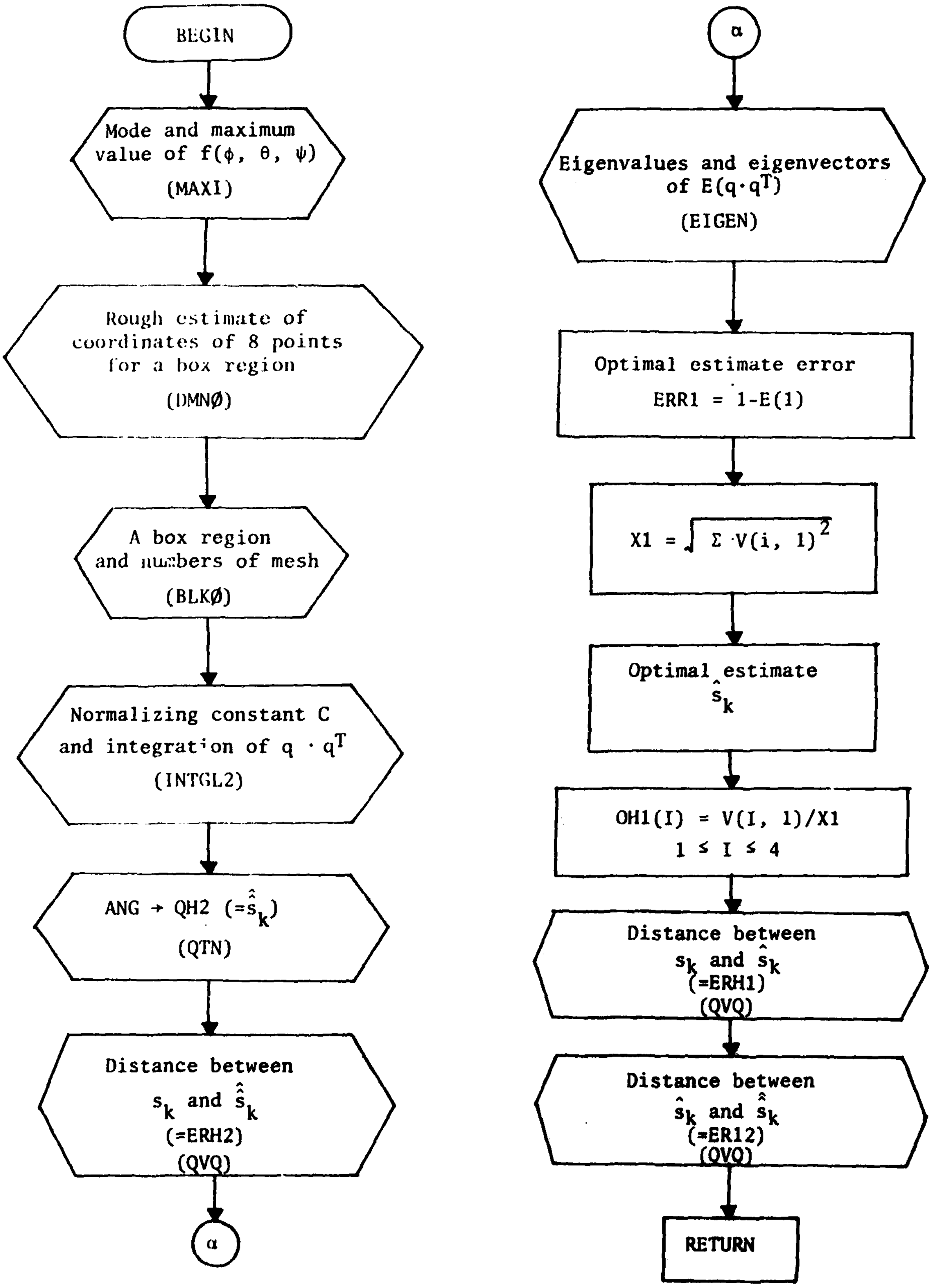
Special Considerations:

1. To find a mode of the conditional density function, a searching method is taken.
2. To find eigenvalues and eigenvectors of a symmetric matrix, $E(q\cdot q^T)$, the matrix is first tridiagonalized by 'TRIDMX' and then eigenvalues by 'EIGVAL', eigenvectors by 'EIGVEC' are obtained.

Other Subprograms Called:

MAXI, DMNØ, BLKØ INTGL2, QTN, QVQ, EIGEN

Flow Chart


BEGIN
Mode and maximum value of $f(\phi, \theta, \psi)$
(MAXI)
Rough estimate of coordinates of 8 points for a box region
(DMNØ)
A box region and numbers of mesh
(BLKØ)
Normalizing constant C and integration of $q \cdot q^T$
(INTGL2)
ANG → QH2 $(=\hat{\hat{s}}_k)$
(QTN)
Distance between s_k and $\hat{\hat{s}}_k$
(=ERH2)
(QVQ)
α
α
Eigenvalues and eigenvectors of $E(q \cdot q^T)$
(EIGEN)
Optimal estimate error
ERR1 = 1-E(1)
$X1 = \sqrt{\Sigma \cdot V(i, 1)^2}$
Optimal estimate $\hat{s}_k$
OH1(I) = V(I, 1)/X1
$1 \leq I \leq 4$
Distance between s_k and $\hat{s}_k$
(=ERH1)
(QVQ)
Distance between $\hat{s}_k$ and $\hat{\hat{s}}_k$
(=ER12)
(QVQ)
RETURN

FR

A subprogram

Purpose:

To compute n! for a given non-negative integer n.

Calling sequence:

```
Y = FR(N).
```

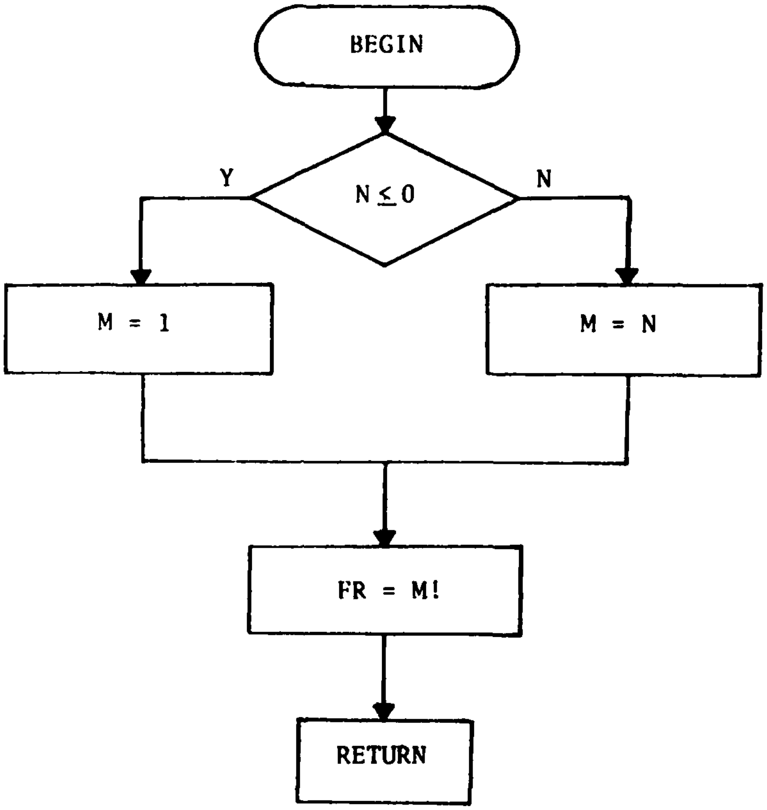
Input:

N: integer

Special Consideration:

1. !f n ≤ 0, N assumes 0

G.5. Flow Chart



FU

A subprogram

Purpose:

To compute the values

$$\begin{bmatrix} -\underline{u}_1(s) \\ \underline{u}_3(s) \\ -\underline{\underline{u}}_1(s) \\ \underline{\underline{u}}_3(s) \\ \underline{u}_2(s) \\ \underline{\underline{u}}_2(s) \end{bmatrix} = \begin{bmatrix} FU(1,\cdot,\cdot,\cdot) \\ FU(2,\cdot,\cdot,\cdot) \\ FU(3,\cdot,\cdot,\cdot) \\ FU(4,\cdot,\cdot,\cdot) \\ FU(5,\cdot,\cdot,\cdot) \\ FU(6,\cdot,\cdot,\cdot) \end{bmatrix}$$

where $\underline{u}_j(s)$ (or $\underline{u}_j(s)$) is given by

$$u_j(5) = m_{j3}a_3(k)D^1_{00}(s) + [\sqrt{2}m_{j2}a_1(k)\mathrm{Re}D^1_{-10}(s)$$

$$+ (m_{j2}a_2(k) - m_{j1}a_1(k))\mathrm{Re}D^1_{-11}(s) - \sqrt{2}M_{j1}a_3(k)\mathrm{Re}D^1_{01}(s)$$

$$+ (m_{j1}a_1(k) + m_{j2}a_2(k))\mathrm{Re}D^1_{11}(s)]$$

$$+ [\sqrt{2}M_{j3}a_2(k)\mathrm{Im}D^1_{-10}(s) - (m_{j1}a_2(k) + m_{j2}a_1(k))\mathrm{Im}D^1_{-11}(s)$$

$$- \sqrt{2}M_{j2}a_3(k)\mathrm{Im}D^1_{01}(s) + (m_{j2}a_1(k) - m_{j1}a_2(k))\mathrm{Im}D^1_{11}(s)]$$

Calling Sequence:

```
Y = FU(K, DD, AA, BB)
```

Common variables:

None.

Input:

K integer. FU = $-\underline{u}_1$ if K=1, FU = $\underline{u}_3$ if K=2

FU = $-\underline{\underline{u}}_1$ if K=3, FU = $\underline{\underline{u}}_3$ if K=4

FU = $\underline{u}_2$ if K=5, FU = $\underline{\underline{u}}_2$ if K=6

DD values $\{ReD^1_{mn}(s)\}$, $\{ImD^1_{mn}(s)\}$, computed in 'DRI2', 3x3 dimensional array.

AA, BB coefficients of $\{ReD^1_{mn}(s)\}$ and $\{ImD^1_{mn}(s)\}$, respectively, computed in 'COEFF1', 6x3x3 dimensional arrays.

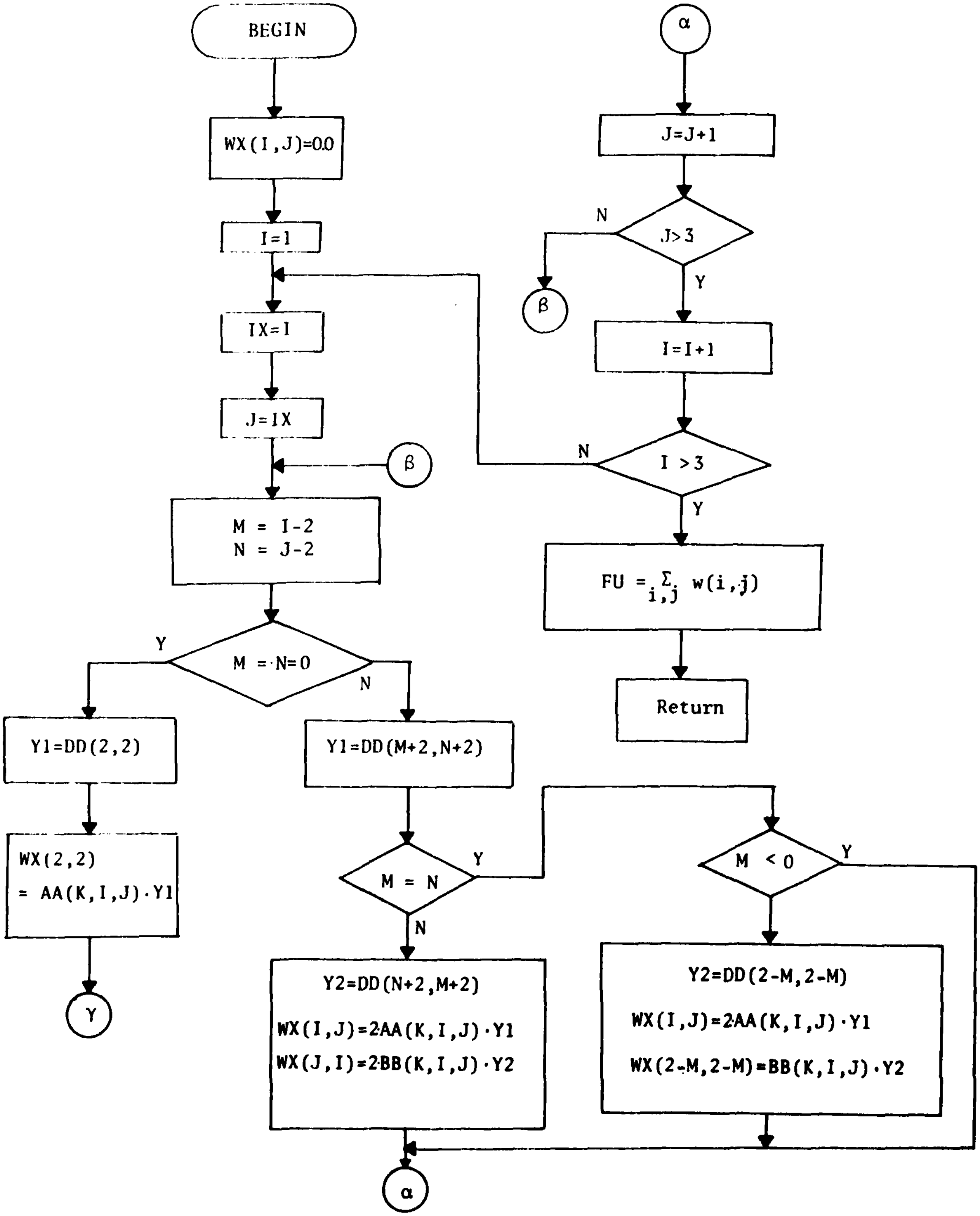
Special Considerations:

None.

Other Subprogram Called:

None

Flow Chart


BEGIN
WX(I,J)=0.0
I=1
IX=1
J=IX
β
M = I-2
N = J-2
M = N=0
Y
N
Y1=DD(2,2)
WX(2,2)
= AA(K,I,J)·Y1
Y
Y1=DD(M+2,N+2)
M = N
Y
N
Y2=DD(N+2,M+2)
WX(I,J)=2AA(K,I,J)·Y1
WX(J,I)=2BB(K,I,J)·Y2
α
M < 0
Y
Y2=DD(2-M,2-M)
WX(I,J)=2AA(K,I,J)·Y1
WX(2-M,2-M)=BB(K,I,J)·Y2
α
J=J+1
J>3
N
Y
β
I=I+1
I >3
N
Y
FU = $\sum_{i,j} w(i,j)$
Return

GAUSS

A subprogram

Purpose:

To generate a random number from a given normal distribution N(m, s) (The program is taken from IBM 360 scientific subroutine package,p.77).

Calling Sequence:

```
CALL GAUSS(IX, S, AM, V)
```

Common Variables:

None

Input:

IX initial random integer to create a sequence of random numbers from the uniform distribution.

S standard deviation.

AM mean

Output:

V random number from the normal distribution.

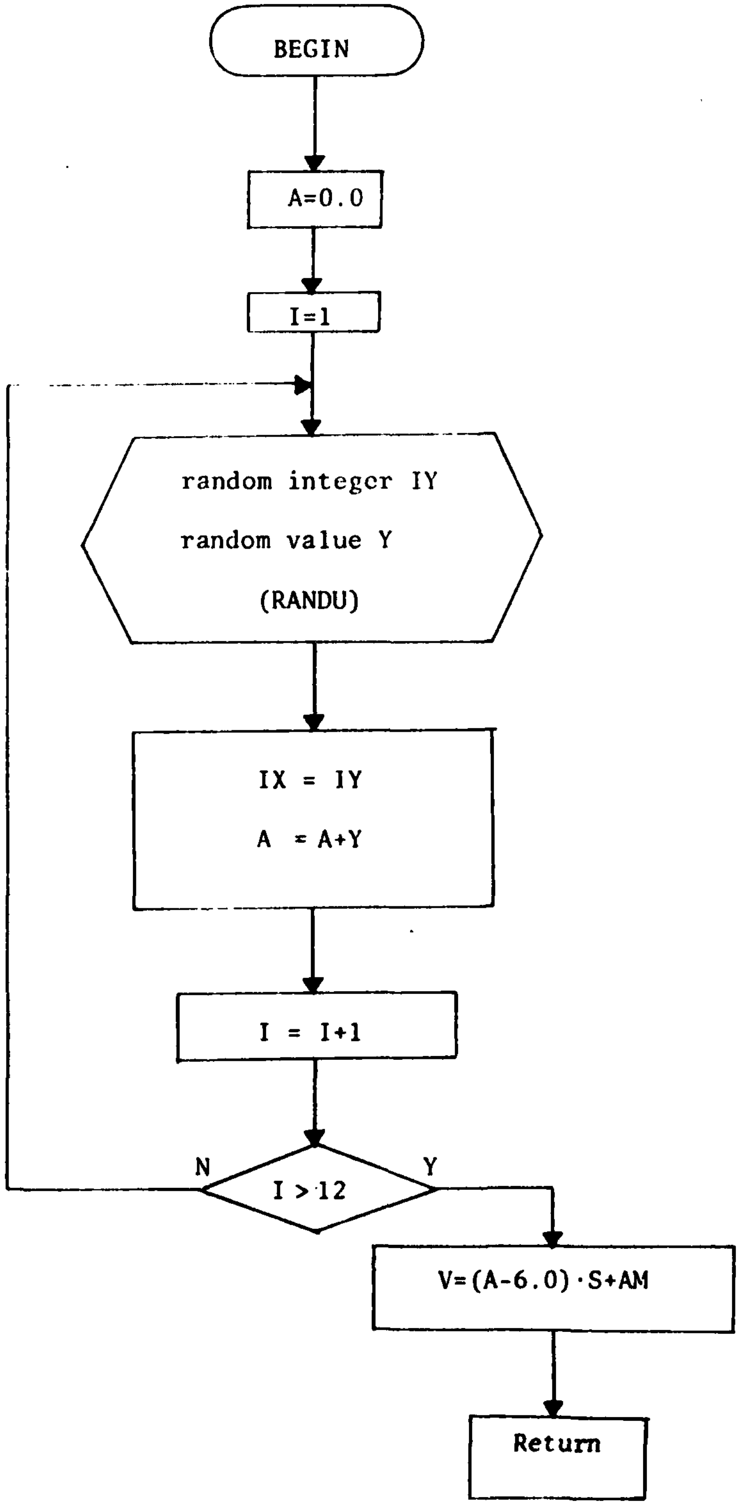
Special Consideration:

1. In the program, the mean is always assumed to be zero.

Other Subprograms Called:

RANDU

Flow Chart


BEGIN
A=0.0
I=1
random integer IY
random value Y
(RANDU)
IX = IY
A = A+Y
I = I+1
N
I > 12
Y
V=(A-6.0)·S+AM
Return

GAUSS4

A subprogram

Purpose:

To generate four independent random values from normal distributions.

Calling Sequence:

```
CALL    GAUSS4(M, IXO, R, V)
```

Common Variables:

None.

Input:

M — total number of times to use RANDU to generate random values from the uniform distribution.

IXO — initial integer for RANDU.

R — 1/R(i) representing the variance of the i^{th} normal distribution, $1 \leq \quad \leq 4$.

Output:

V — random values, 4-dim. array.

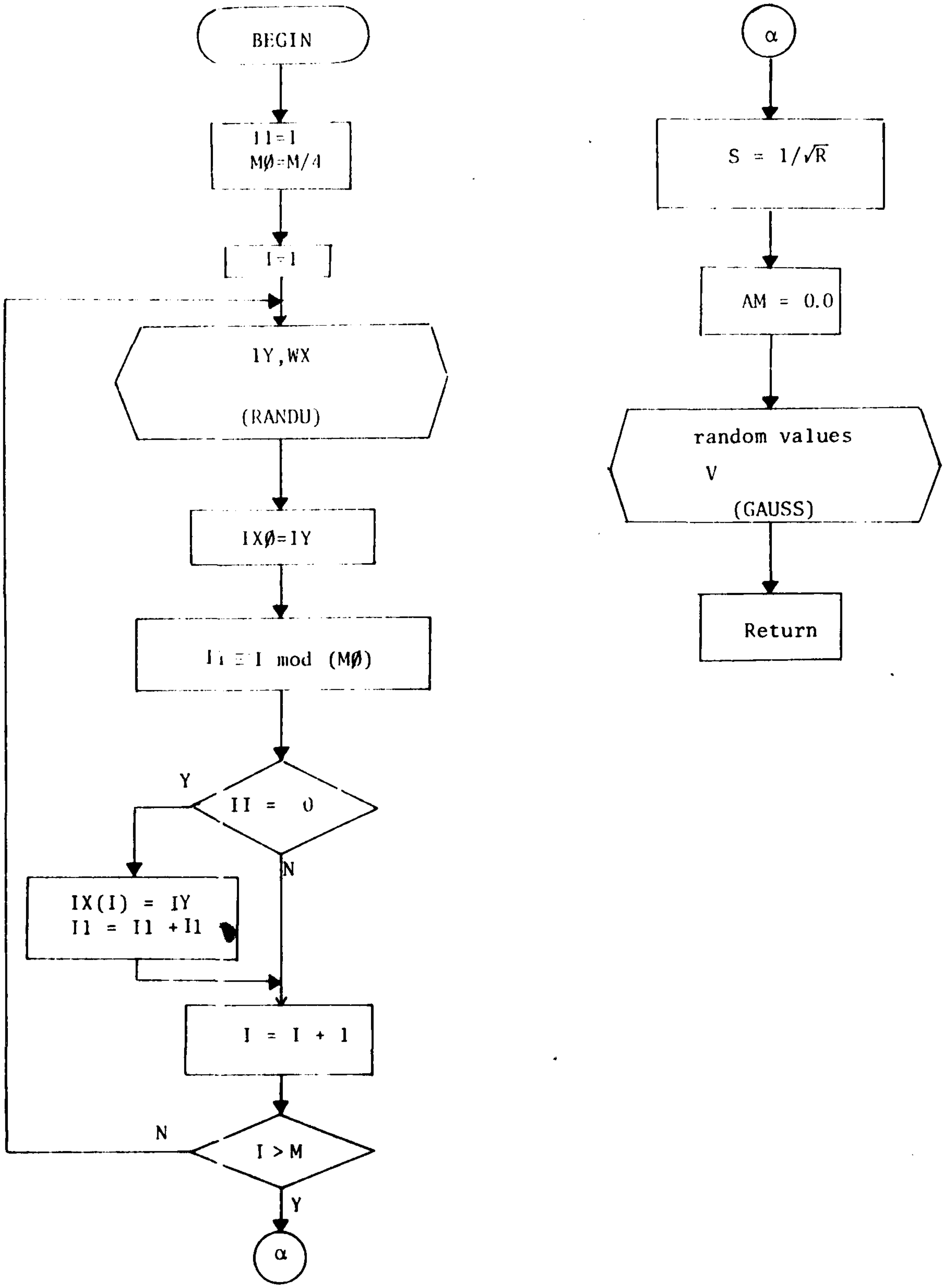
Special Considerations:

1. Four random integers are chosen among M integers for V(1), V(2), V(3), V(4).

Other Subprograms Called:

RANDU, GAUSS.

Flow Chart


BEGIN
I1=1
MØ=M/4
I=1
IY,WX
(RANDU)
IXØ=IY
I1 ≡ I mod (MØ)
Y
II = 0
N
IX(I) = IY
I1 = I1 +I1
I = I + 1
N
I > M
Y
α
α
S = 1/√R
AM = 0.0
random values
V
(GAUSS)
Return

INITPQ

A Subprogram

Purpose:

To give an initial density function for the conditional density functions $\{P(s_k|m^k)\}$ i.e $P(s_o|m^o) = P(s_o)$.

As an initial density $P(s_o)$, the following type of distribution on SO(3) is chosen,

$$P(s_o) = C\cdot\exp\,(r(a_1q_1 + a_2q_2 + a_3q_3 + a_4q_4)^2)$$

where (a_1, a_2, a_3, a_4) is a constant quaternion, representing the mode, and $s_o = (q_1, q_2, q_3, q_4)$.

In the program $(a_1q_1 + a_2q_2 + a_3q_3 + a_4q_4)^2$ is expressed as an expansion of $\{ReD^1_{mn}\}$ $\{ImD^1_{mn}\}$, and the coefficients of the expansion are stored in (PØ, QØ).

Calling Sequence:

```
CALL INITPQ (AK, A, PØ, QØ)
```

Common Variables:

None

Input:

AK parameter representing the magnitude of mode of the distribution, r.

A 4 dimensional array, mode (a_1, a_2, a_3, a_4).

Output:

PØ, QØ 2x5x5 dimensional arrays, coefficients of $\{ReD^\ell_{mn}(s_o)\}$, $\{ImD^\ell_{mn}(s_o)\}$, respectively.

Special Considerations:

1. The relations between (q_1, q_2, q_3, q_4) and $\{ D^1_{mn}(s_o)\}$ are given

$$q_1^2 = \frac{1}{4} - (\frac{1}{4})D^1_{00} - (\frac{1}{2})ReD^1_{-11}$$

$$q_2^2 = \frac{1}{4} - (\frac{1}{4})D^1_{00} + (\frac{1}{2})ReD^1_{-11}$$

$$q_3^2 = \frac{1}{4} + (\frac{1}{4})D^1_{00} - (\frac{1}{2})ReD^1_{11}$$

$$q_4^2 = \frac{1}{4} + (\frac{1}{4})D^1_{00} + (\frac{1}{2})ReD^1_{11}$$

$$q_1q_2 = -(\frac{1}{2})ImD^1_{-11},\ q_1q_3 = \frac{\sqrt{2}}{4}(ReD^1_{-10} - ReD^1_{01})$$

$$q_1q_4 = \frac{\sqrt{2}}{4}(ImD^1_{-10} + ImD^1_{01}),\ q_2q_3 = -\frac{\sqrt{2}}{4}(ImD^1_{01} - ImD^1_{-10})$$

$$q_2q_4 = \frac{\sqrt{2}}{4}(ReD^1_{-10} + ReD^1_{01})$$

2. In the program, the north pole (0, 0, 0, 1) is taken as a mode.

3. Coefficients of $\{ReD^1_{mn}\}$ and $\{ImD^1_{mn}\}$ are stored in PØ(1,·,·) and QØ(1, ·, ·) respectively. By the properties of

$$ReD^{\ell}_{-m-n} = (-1)^{n-m}ReD^{\ell}_{mn}$$

$$ImD^{\ell}_{-mn} = (-1)^{n-m+1}ImD^{\ell}_{mn},$$

The coefficients are arranged using the following signs tables:

$$\begin{bmatrix} 1 & 1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \end{bmatrix} \quad \text{for real parts}$$

$$\begin{bmatrix} -1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$ for imaginary parts

Example: If the coefficient of ReD^{1}_{01} is

$\frac{\sqrt{2}}{2}\,(-a_1a_3 + a_2a_4)\cdot r$, then

$$P\emptyset(1, 2, 3) = \frac{\sqrt{2}}{2}\,(-a_1a_3 + a_2a_4)\cdot r$$

$$P\emptyset(1, 4, 2) = \frac{\sqrt{2}}{2}\,(\ a_1a_3 - a_2a_4)\cdot r$$

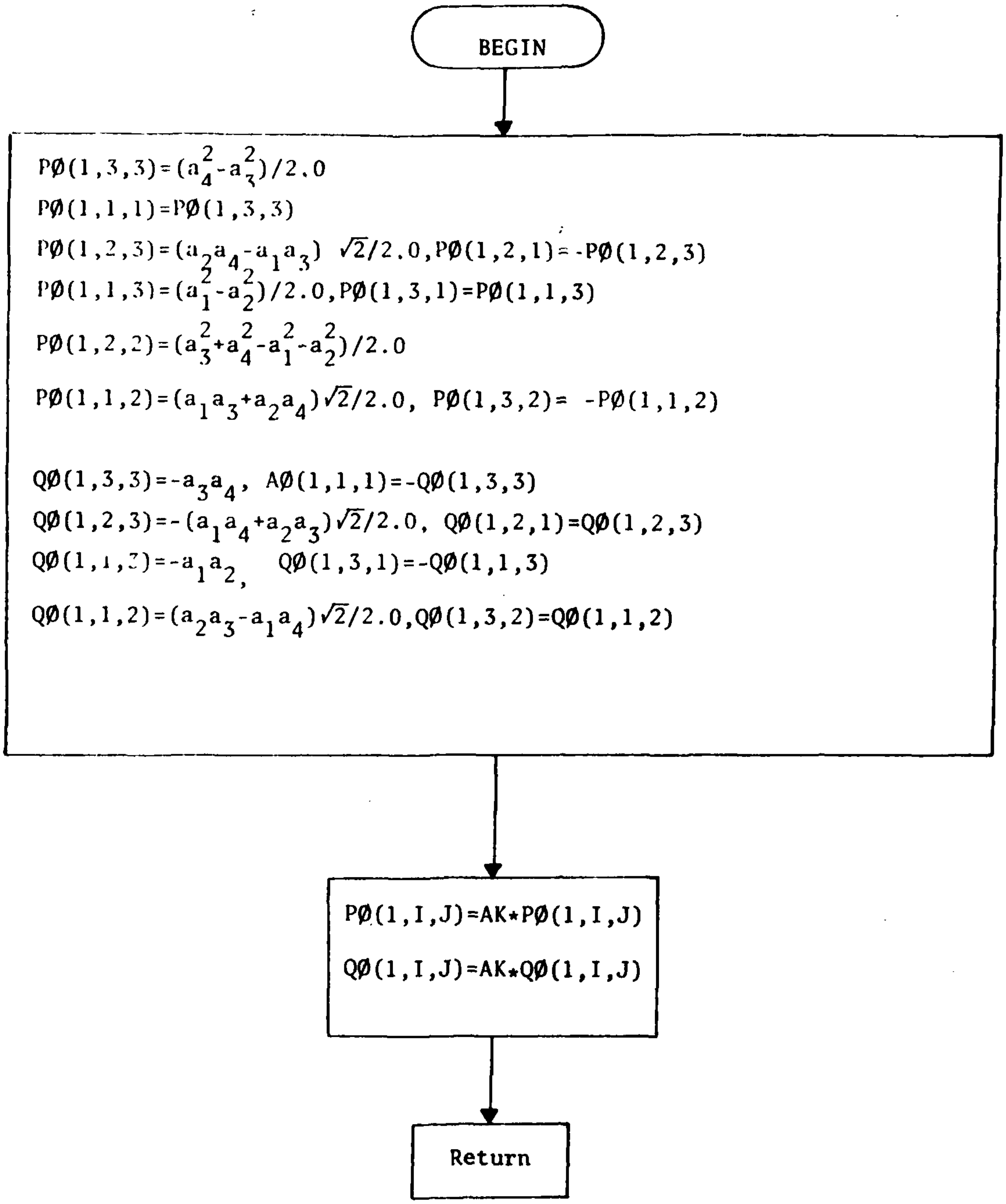
this gives a term

$$\frac{1}{2}P\emptyset(1, 2, 3,)\ ReD^{1}_{01} + \frac{1}{2}P\emptyset(1, 1, 2)ReD^{1}_{0-1}$$

$$= \frac{\sqrt{2}}{4}\,(-a_1a_3 + a_2a_4)\cdot r\ ReD^{1}_{01} + \frac{\sqrt{2}}{4}(-a_1a_3 + a_2a_4)\cdot r\ ReD^{1}_{0-1}$$

$$= \frac{\sqrt{2}}{2}\,(-a_1a_3 + a_2a_4)\cdot r\ ReD^{1}_{01}\ \text{(since } ReD^{1}_{0-1} = -ReD^{1}_{01}\text{)}$$

Flow Chart



PØ(1,3,3)=$(a_4^2-a_3^2)/2.0$

PØ(1,1,1)=PØ(1,3,3)

PØ(1,2,3)=$(a_2a_4-a_1a_3)\ \sqrt{2}/2.0$,PØ(1,2,1)=-PØ(1,2,3)

PØ(1,1,3)=$(a_1^2-a_2^2)/2.0$,PØ(1,3,1)=PØ(1,1,3)

PØ(1,2,2)=$(a_3^2+a_4^2-a_1^2-a_2^2)/2.0$

PØ(1,1,2)=$(a_1a_3+a_2a_4)\sqrt{2}/2.0$, PØ(1,3,2)= -PØ(1,1,2)

QØ(1,3,3)=$-a_3a_4$, AØ(1,1,1)=-QØ(1,3,3)

QØ(1,2,3)=$-(a_1a_4+a_2a_3)\sqrt{2}/2.0$, QØ(1,2,1)=QØ(1,2,3)

QØ(1,1,3)=$-a_1a_2$, QØ(1,3,1)=-QØ(1,1,3)

QØ(1,1,2)=$(a_2a_3-a_1a_4)\sqrt{2}/2.0$,QØ(1,3,2)=QØ(1,1,2)

PØ(1,I,J)=AK*PØ(1,I,J)

QØ(1,I,J)=AK*QØ(1,I,J)

INSDD

A subprogram

Purpose:

To generate a table for $\{d^{\ell}_{mn}(\theta)\}$, where $1 \leq \ell \leq 2$, $-\ell \leq m,n \leq \ell$

Calling Sequence:

```
CALL INSDD (MT, SDD1, SDD2)
```

Common Variables:

D

Input:

D mesh size, i.e. D = π/MT, given in the main program.

MT number of meshes in[0,π] (in the program MT=16).

Output:

SDD1 3x3x17 dimensional array, $\{d^{\ell}_{mn}(\theta_i)\}$

SDD2 5x5x17 dimensional array, $\{d^{\ell}_{mn}(\theta_i)\}$,
where $\theta_i = (\pi/MT)*(i-1)$, i=1,2, ..., (MT+1)

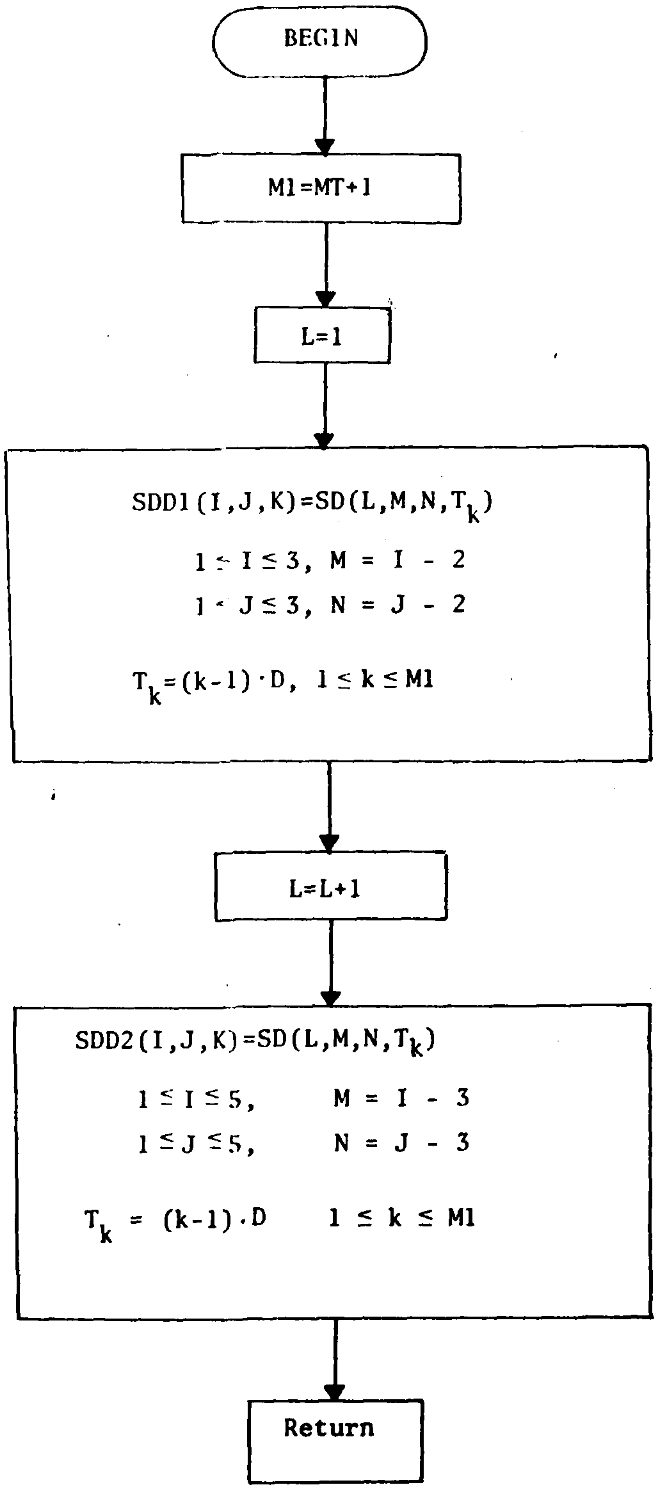
Special Considerations

1. The values $d^1_{mn}(0_i)$, $d^2_{mn}(\theta_i)$ are stored in SDD1(m+2, n-2)
SDD1(m+2, n+2, I) and SDD2(m+3, n+3, I), respectively.

Other Subprograms Called:

SD.

Flow Chart


BEGIN
M1=MT+1
L=1
SDD1(I,J,K)=SD(L,M,N,T_k)
$1 \le I \le 3$, M = I - 2
$1 \le J \le 3$, N = J - 2
T_k=(k-1)·D, $1 \le k \le$ M1
L=L+1
SDD2(I,J,K)=SD(L,M,N,T_k)
$1 \le I \le 5$, M = I - 3
$1 \le J \le 5$, N = J - 3
T_k = (k-1)·D $1 \le k \le$ M1
Return

INTGLE

A Subprogram

Purpose:

To compute the expectation $E(q \cdot q^T)$ with respect to the conditional density

$$P(s_k | m^k) = C \cdot \exp\ (f(s_k))$$

where

$$f(s_k) = \sum_{\ell=1}^{2} [\tfrac{1}{2} P^{\ell k}_{100} D_{00}(s_k) + \sum_{\substack{m,n=-\ell \\ m<n \\ \text{or} \\ m=n>0}}^{\ell} (P^{\ell k}_{1mn} \mathrm{Re} D^{\ell}_{mn}(s_k) + P^{\ell k}_{2mn} \mathrm{Im} D^{\ell}_{mn}(s_k))]$$

and C is the normalizing constant, $q = (q_1, q_2, q_3, q_4)$ quaternion variable.

Calling Sequence:

```
CALL INTGLE(IND, SDD1, SDD2, SINE, COSN, P, Q, FMAX, ANG, C, QQ).
```

Common Variables:

MT, NZ, H, D

Input:

MT	number of meshes in $[0,\pi]$ for θ
NZ	number of meshes in $[0, 2\pi]$ for ϕ and ψ
D	mesh size for θ , i.e. $D = \pi / MT$
H	mesh size for ϕ and ψ , i.e. $H=2\pi/NZ$ The above values are computed in the main program for integration.

SDD1, SDD2 3x3x17 and 5x5x17 dimensional array, representing $\{ d^1_{mn}(\theta_i) \}$, $\{d^2_{mn}(\theta_i)\}$respectively, where θ_i = D·(i-1), i=1, 2, ..., (MT+1).

IND integer IND=0 if max $(f(s_k))$(=FMAX) on SØ(3) to be computed. IND=1 if a normalizing constant to be computed and IND=2 if $E(q\cdot q^T)$ to be computed.

SINE, COSN 33-dimensional arrays, tables of $\{\sin(\phi_i)\}$, $\{\cos(\phi_i)\}$, respectively, where ϕ_i = H·(i-1), i=1, 2, ...,(NZ+1).

P, Q 2x5x5 dimensional arrays representing $\{P^{\ell k}_{1mn}\}$, $\{P^{\ell k}_{2mn}\}$ respectively.

Both are computed in 'NPAQ'.

Output:

FMAX MAX(f(s)) on SO(3)

ANG 3-dimensional array representing the mode of f(s) in terms of Euler angles.

C normalizing constant

QQ 4x 4 dim array, $E(q\cdot q^T)$.

Special Considerations

1. In order to avoid overflow of computation $\exp(f(\phi, \theta, \psi))$, $f(\phi,\theta,\psi)$ is replaced by $f - f_o$, where f_o = FMAX-165.0 if FMAX ≦ 165.0 , and $f_o = 0$ if FMAX 165.0.

2. Normalizing constant C is computed to satisfy

$$\int C \exp(f(s)-f_o)\, ds = 1$$

3. The mode of f at $(k-1)^{th}$ stage will be used to determine a predicted estimate of s_k.

4. $q \cdot q^T = \begin{bmatrix} q_1^2 & q_1q_2 & q_1q_3 & q_1q_4 \\ q_2q_1 & q_2^2 & q_2q_3 & q_2q_4 \\ q_3q_1 & q_3q_2 & q_3^2 & q_3q_4 \\ q_4q_1 & q_4q_2 & q_4q_3 & q_4^2 \end{bmatrix}$ is symmetric,

only the upper triangle part is computed.

5. For any values involving ReD^{ℓ}_{mn}, ImD^{ℓ}_{mn}, sin ϕ, cos ϕ, tables SDD1, SDD2, SINE,COSN are used instead of calling subprograms. For example, to compute

$$\mathrm{ReD}^1_{mn}\,(\phi_j,\theta_i,\psi_k) = \cos\,(\tfrac{\pi}{2}\,(m-n) - (m\phi_j + n\psi_k))d^1_{mn}(\theta_i)$$

an integer MN1 is computed, so that $\mathrm{ReD}^1_{mn}(\phi_j,\theta_i,\psi_k) =$ COSN(MN1)·SDD1(M + 2,N+2, I). MN1 is determined using the following relations,

$$\frac{\pi}{2} \leftrightarrow \frac{NZ}{4}$$

$$m\phi_j+n\psi_k \leftrightarrow N(J-1) + N(K-1)$$

$$\begin{aligned} \tfrac{\pi}{2}(m-n) - (m\phi_j+n\psi_k) \leftrightarrow\ & (\tfrac{NZ}{4})(M-N) - M(J-1) - N(K-1) \\ & \equiv MN \ (\mathrm{mod}\ (NZ)) \\ & = MN \quad \text{if } MN \geq 0 \\ & \ MN+NZ \quad \text{if } MN < 0 \end{aligned}$$

$$\Rightarrow MN1 = MN + 1$$

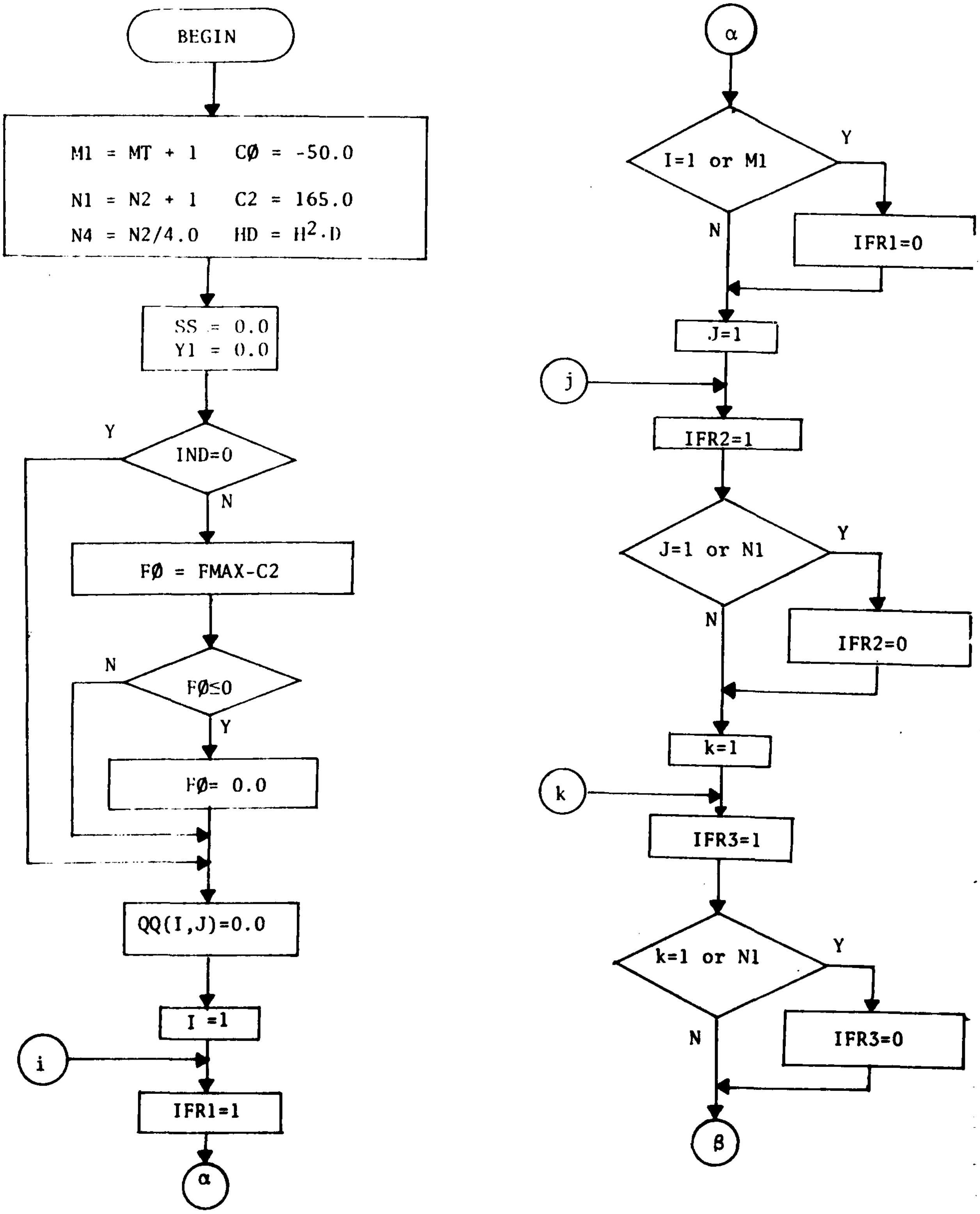
6. $q \cdot q^T$ is expressed as a function matrix of Euler angles

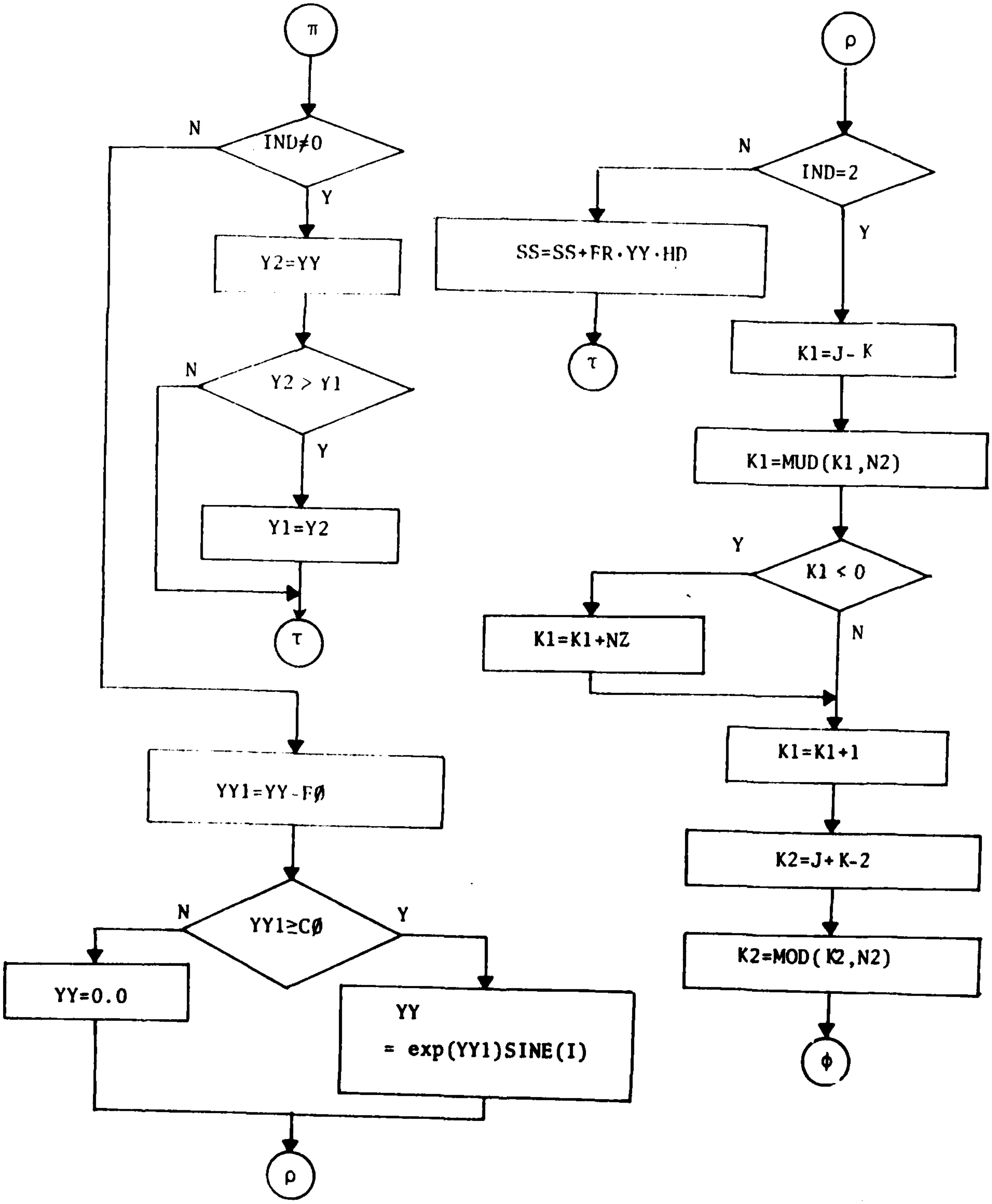
$$\begin{bmatrix} (1-\cos\theta)(1+\cos(\phi-\psi)/4.0 & & & \\ (1-\cos\theta)\sin(\phi-\psi)/4.0 & (1-\cos\theta)(1-\cos(\phi-\psi) & & \\ \sin\theta(\sin\phi+\sin\psi)/4.0 & \sin\theta(\cos\psi-\cos\phi)/4.0 & (1+\cos\theta)(1-\cos(\phi+\psi))/4.0 & \\ \sin\theta(\cos\psi-\cos\phi)/4.0 & \sin\theta(\sin\phi-\sin\psi)/4.0 & (1+\cos\theta)\sin(\phi+\psi)/4.0 & (1+\cos\theta)(1+\cos(\phi-\psi)) \end{bmatrix}$$

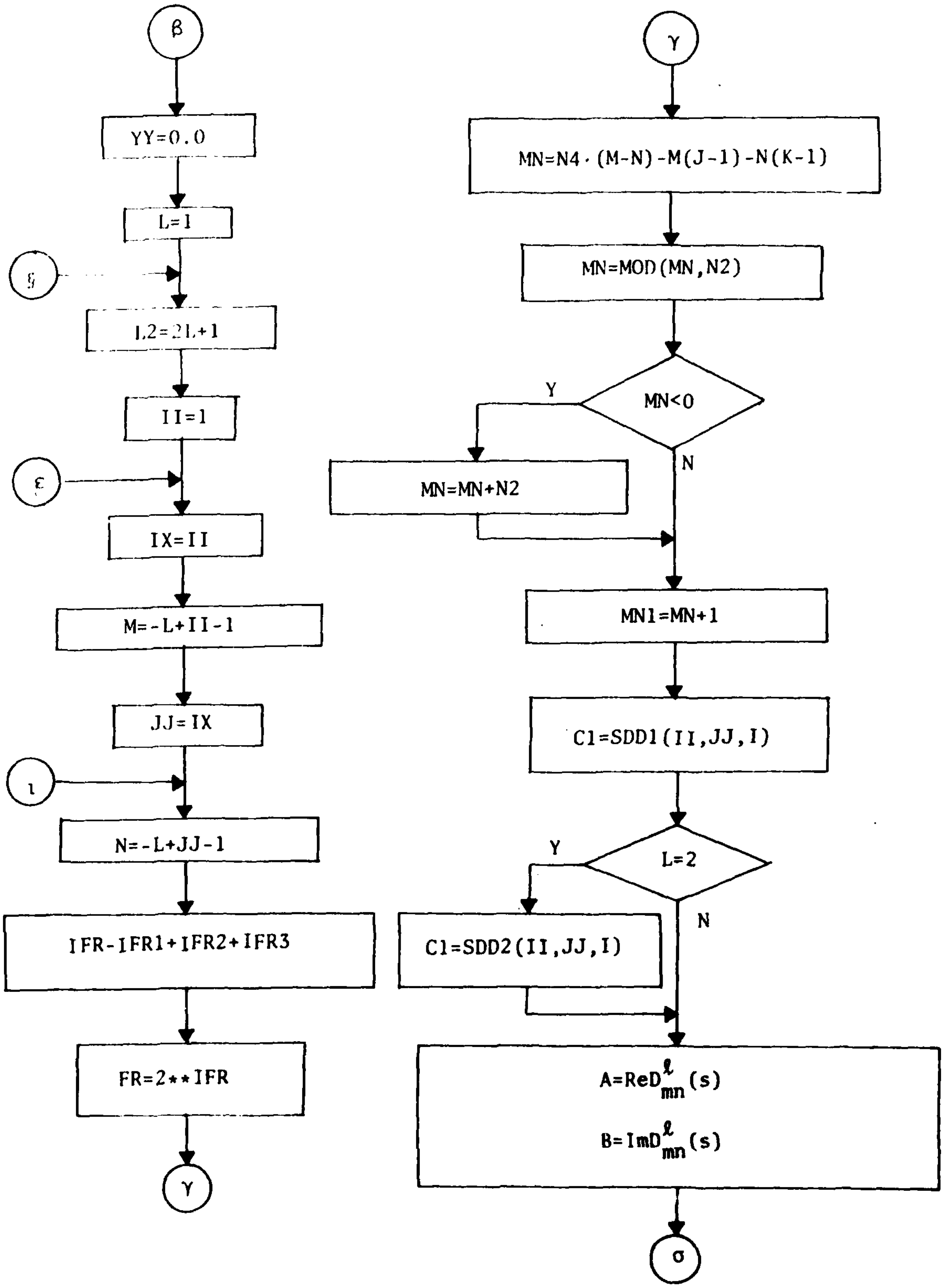
where the entries of the upper triangle are omitted.

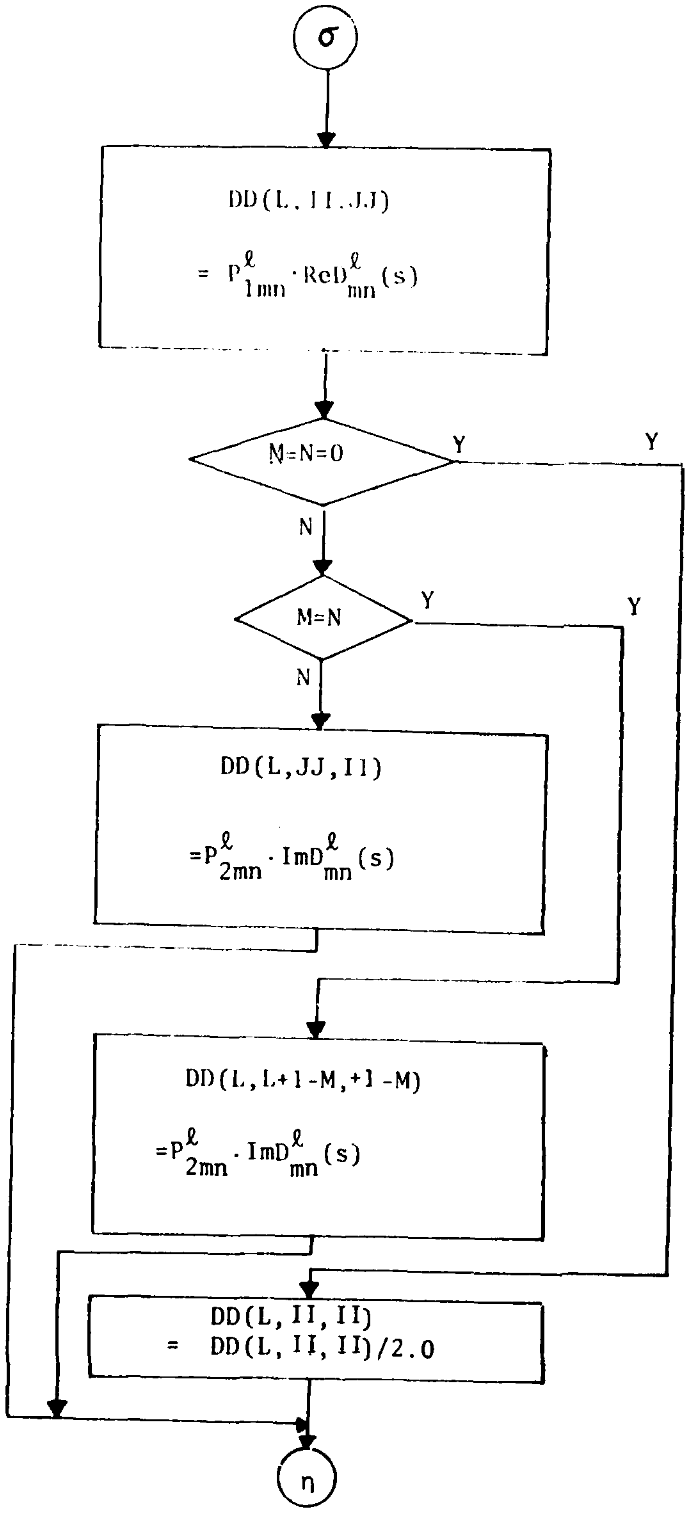
7. In the computation of exp $(f-f_o)$, exp $(f-f_o) \equiv 0$ if $f-f_o \leq -50.0$ to avoid underflow .

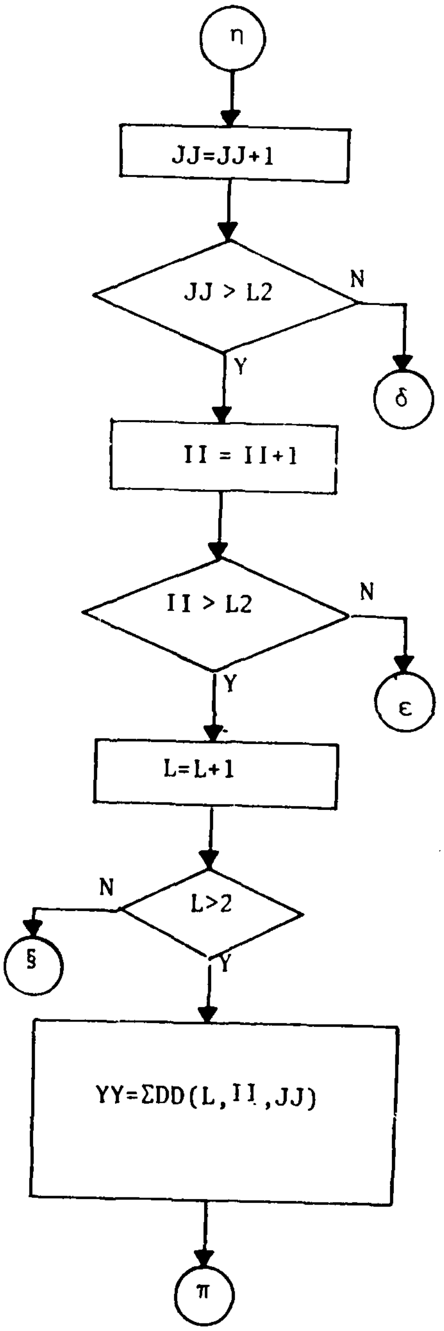
Flow Chart

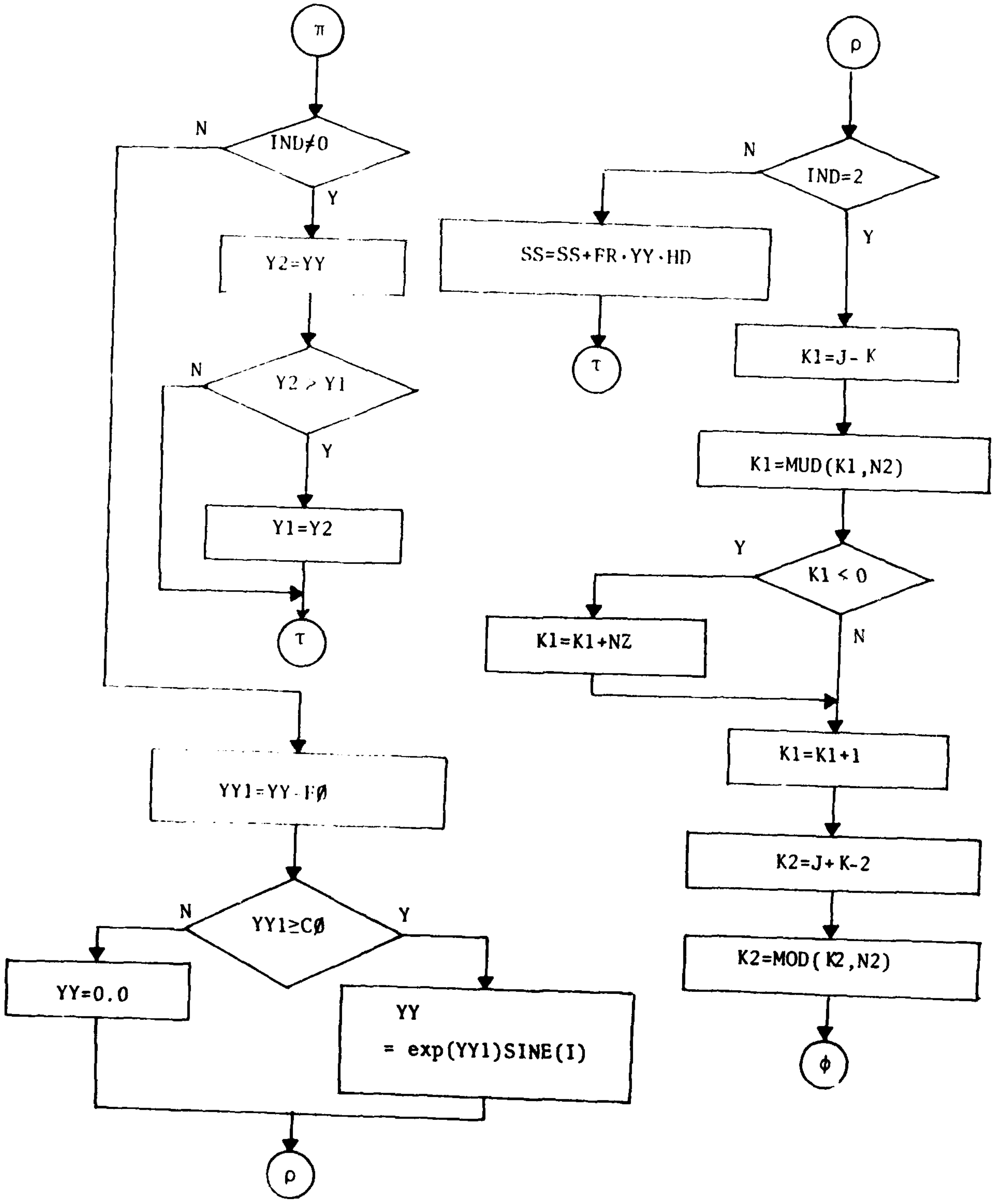

BEGIN
M1 = MT + 1
CØ = -50.0
N1 = N2 + 1
C2 = 165.0
N4 = N2/4.0
HD = $H^2 \cdot D$
SS = 0.0
Y1 = 0.0
IND=0
Y
N
FØ = FMAX-C2
FØ≤0
N
Y
FØ= 0.0
QQ(I,J)=0.0
I =1
i
IFR1=1
α
α
I=1 or M1
Y
N
IFR1=0
J=1
j
IFR2=1
J=1 or N1
Y
N
IFR2=0
k=1
k
IFR3=1
k=1 or N1
Y
N
IFR3=0
β

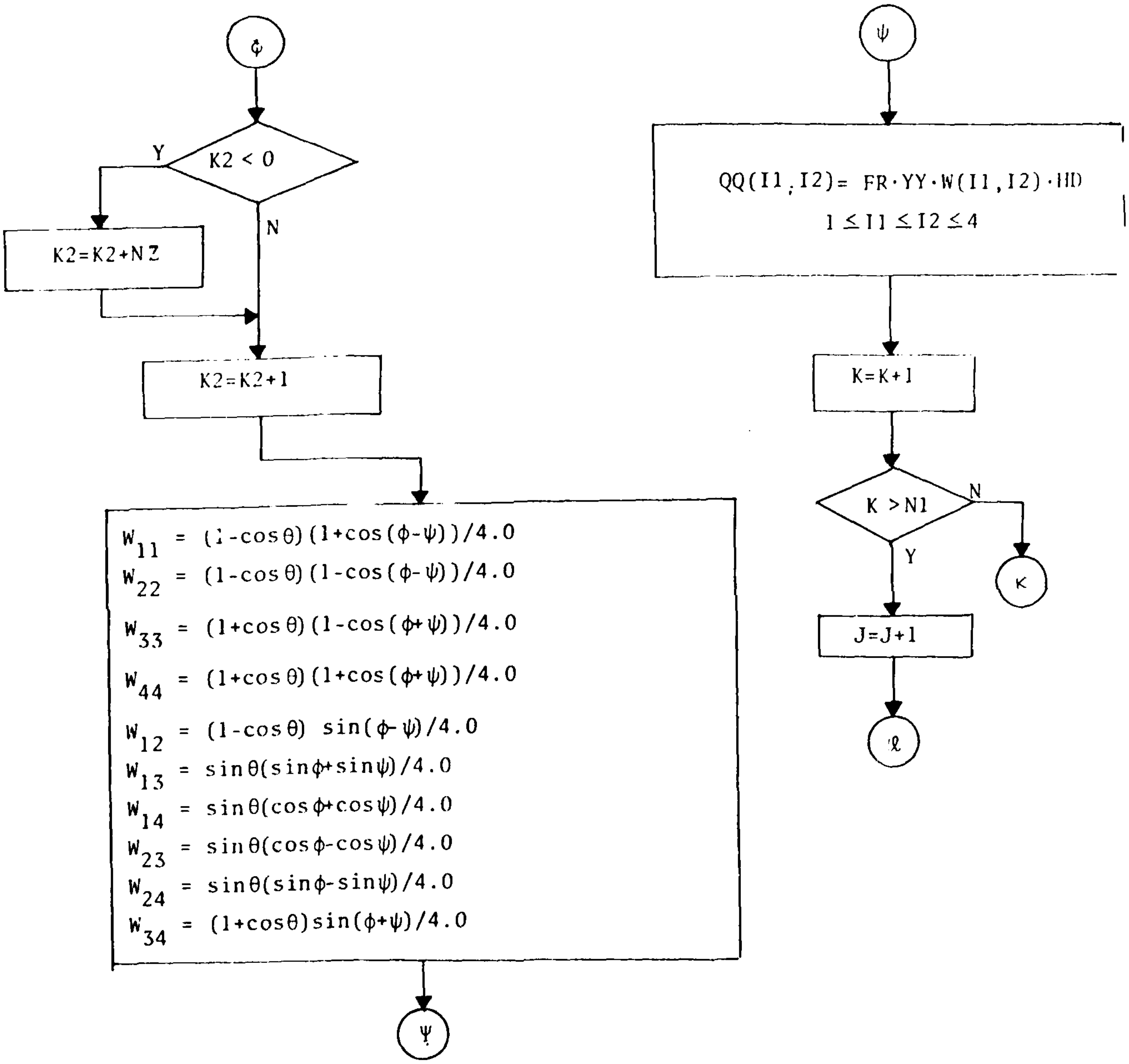

π
N
IND≠0
Y
Y2=YY
N
Y2 > Y1
Y
Y1=Y2
τ
YY1=YY-FØ
N
YY1≥CØ
Y
YY=0.0
YY
= exp(YY1)SINE(I)
ρ
ρ
N
IND=2
Y
SS=SS+FR·YY·HD
τ
K1=J- K
K1=MUD(K1,N2)
Y
K1 < 0
N
K1=K1+NZ
K1=K1+1
K2=J+K-2
K2=MOD(K2,N2)
φ

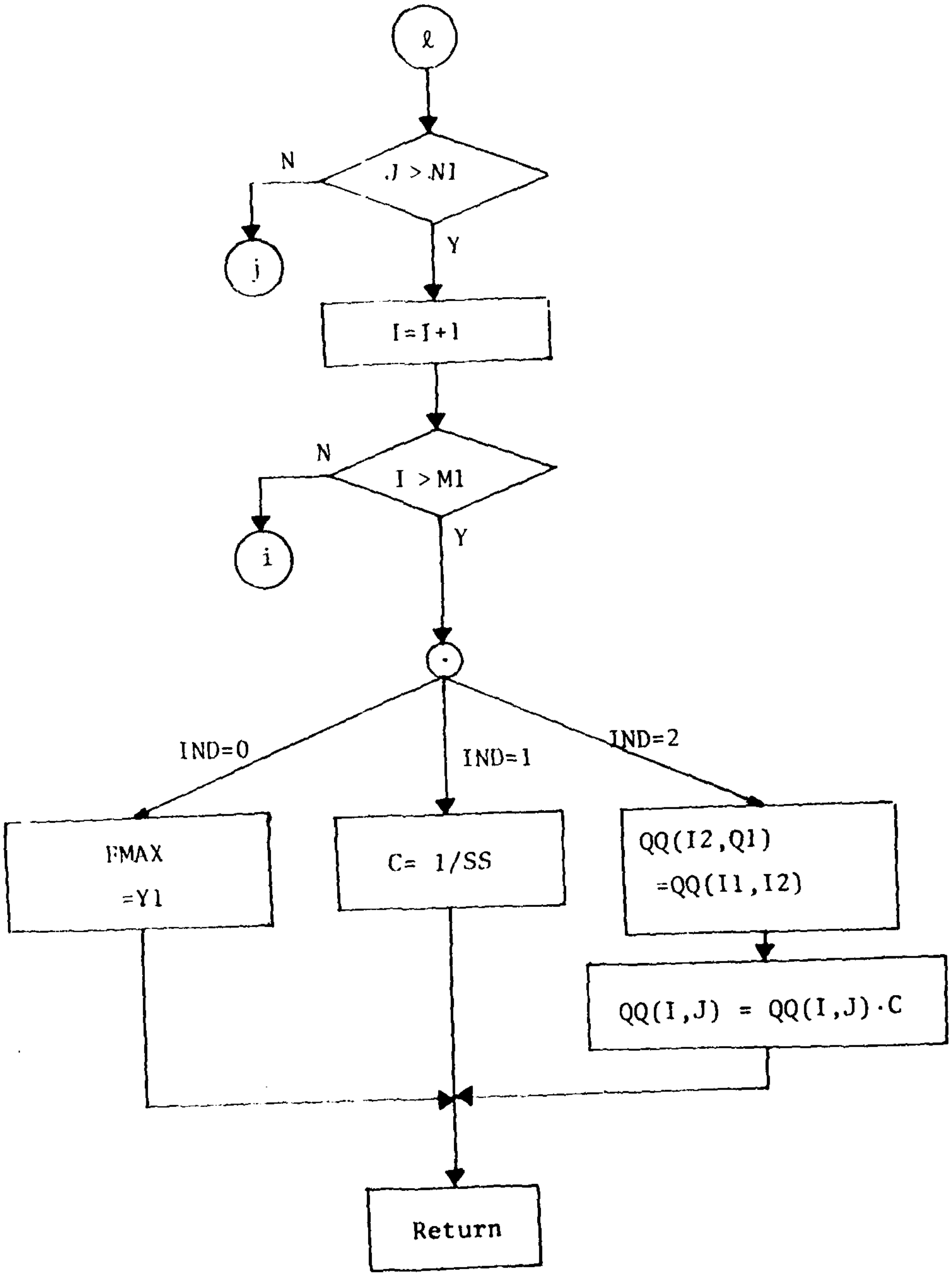

β
YY=0.0
L=1
ξ
L2=2L+1
II=1
ε
IX=II
M=-L+II-1
JJ=IX
ι
N=-L+JJ-1
IFR-IFR1+IFR2+IFR3
FR=2**IFR
γ
γ
MN=N4·(M-N)-M(J-1)-N(K-1)
MN=MOD(MN,N2)
MN<0
Y
N
MN=MN+N2
MN1=MN+1
C1=SDD1(II,JJ,I)
L=2
Y
N
C1=SDD2(II,JJ,I)
A=ReD$^{\ell}_{mn}$(s)
B=ImD$^{\ell}_{mn}$(s)
σ


σ
DD(L, II, JJ)
= P^ℓ_1mn · ReD^ℓ_mn(s)
M=N=0
Y
Y
N
M=N
Y
Y
N
DD(L,JJ,II)
=P^ℓ_2mn · ImD^ℓ_mn(s)
DD(L,L+1-M,+1-M)
=P^ℓ_2mn · ImD^ℓ_mn(s)
DD(L, II, II)
= DD(L, II, II)/2.0
η


η
JJ=JJ+1
JJ > L2
N
δ
Y
II = II+1
II > L2
N
ε
Y
L=L+1
L>2
N
§
Y
YY=ΣDD(L,II,JJ)
π


π
IND≠0
N
Y
Y2=YY
Y2 ≥ Y1
N
Y
Y1=Y2
τ
YY1=YY-FØ
YY1≥CØ
N
Y
YY=0.0
YY
= exp(YY1)SINE(I)
ρ
ρ
IND=2
N
Y
SS=SS+FR·YY·HD
τ
K1=J- K
K1=MUD(K1,N2)
K1 < 0
Y
N
K1=K1+NZ
K1=K1+1
K2=J+K-2
K2=MOD(K2,N2)
φ


ϕ
K2 < 0
Y
N
K2=K2+NZ
K2=K2+1
$W_{11} = (1-\cos\theta)(1+\cos(\phi-\psi))/4.0$
$W_{22} = (1-\cos\theta)(1-\cos(\phi-\psi))/4.0$
$W_{33} = (1+\cos\theta)(1-\cos(\phi+\psi))/4.0$
$W_{44} = (1+\cos\theta)(1+\cos(\phi+\psi))/4.0$
$W_{12} = (1-\cos\theta)\ \sin(\phi-\psi)/4.0$
$W_{13} = \sin\theta(\sin\phi+\sin\psi)/4.0$
$W_{14} = \sin\theta(\cos\phi+\cos\psi)/4.0$
$W_{23} = \sin\theta(\cos\phi-\cos\psi)/4.0$
$W_{24} = \sin\theta(\sin\phi-\sin\psi)/4.0$
$W_{34} = (1+\cos\theta)\sin(\phi+\psi)/4.0$
Ψ
ψ
$QQ(I1,I2) = FR\cdot YY\cdot W(I1,I2)\cdot HD$
$1 \leq I1 \leq I2 \leq 4$
K=K+1
K >N1
N
Y
K
J=J+1
ℓ


ℓ
J > N1
N
Y
j
I = I + 1
I > M1
N
Y
i
IND=0
IND=1
IND=2
FMAX
=Y1
C= 1/SS
QQ(I2,Q1)
=QQ(I1,I2)
QQ(I,J) = QQ(I,J)·C
Return

INTGL2

A Subprogram

Purpose:

To compute the normalizing constant C in the conditional density function

$P(s_k|m^k) = C\cdot\exp\ (f(s_k))$ and evaluate

$E(q\cdot q^T)$ with respect to $P(s_k|m^k)$.

In the program, every integration is done locally arround the mode of $P(s_k|m^k)$.

Calling Sequence:

CALL INTGL2 (IND, P, Q, FMAX, A, NZ, NT, NW, HØ, C, QQ)

Common Variables:

PAI2

Input:

PAI2	2π
IND	Integer, IND=1 if a normalizing constant to be computed, IND=2 if $E(q\cdot q^T)$ to be computed.
P, Q	$\{P^{\ell k}_{1mn}\}$, $\{P^{\ell k}_{1mn}\}$, 2x5x5-dim arrays
FMAX	Maximum of f(s) on SO(3), obtained in 'MAXI'
A	Starting point (Z(1), T(1), W(1)) for local integration, 3-dim array
NZ, NT, NW	Numbers of mesh in [Z(1), Z(2)], [T(1), T(2)], [W(1), W(2)], respectively.
HØ	Initial meshsize for integration

Output:

C	Normalizing constant
QQ	$E(q\cdot q^T)$, 4x4-dim array

Special Considerations:

1. In order to avoid overflows of computation exp $(f(\phi, \theta, \psi))$, $f(\phi, \theta, \psi)$ is replaced by $f-f_o$, where f_o = FMAX-165.Ø if FMAX $\geq$ 165.Ø and f_o=Ø, if FMAX $\leq$ 165.Ø. Also to avoid underflows, $\exp(f-f_o)=0.0$, if $f-f_o \leq -50.0$.

2. Normalizing constant C is computed to satisfy

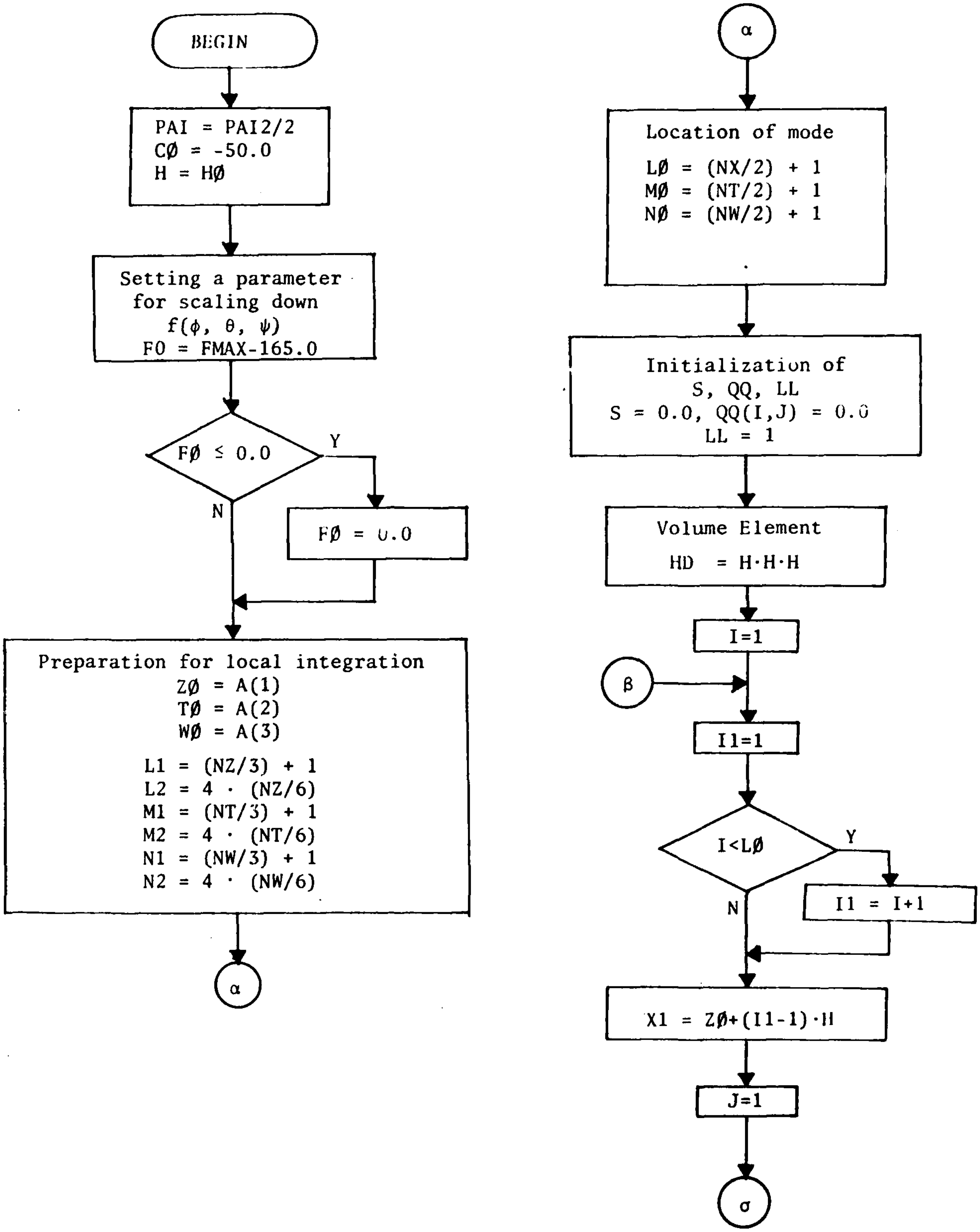
   $$\int C \cdot \exp(f(s)-f_o)\, ds = 1.$$

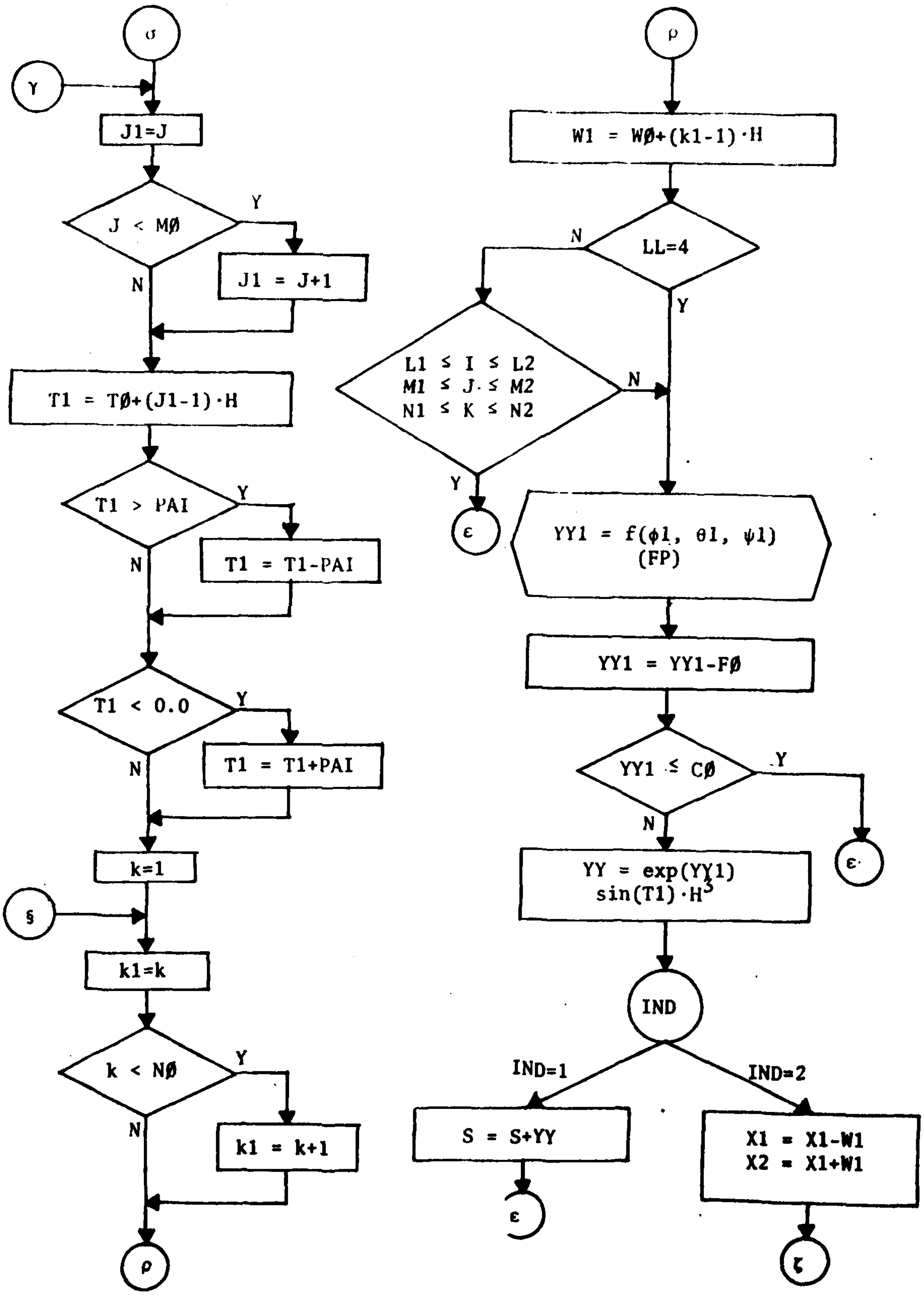
3. Local integrations are done in four steps. At each step except the last one, the Riemann integration is done numerically skipping the 1/3 x 1/3 x 1/3 of the original box region around the mode (center) and the skipped box region will become a new box region for the following step. The stepsize will be one half of the previous one. At the 4th step, the numerical integration is done on the whole box region without skipping the middle region.

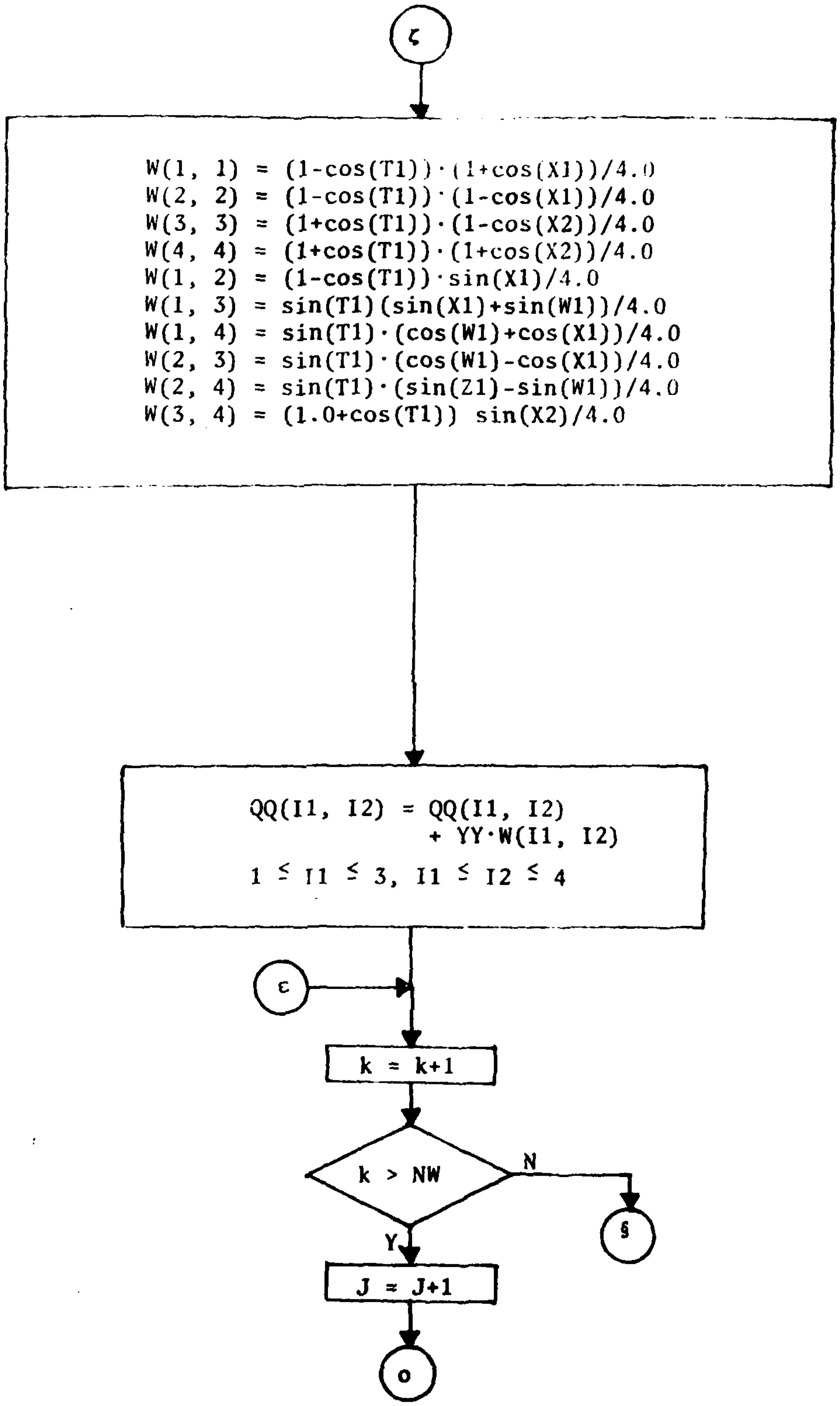
Other Subprograms Called:

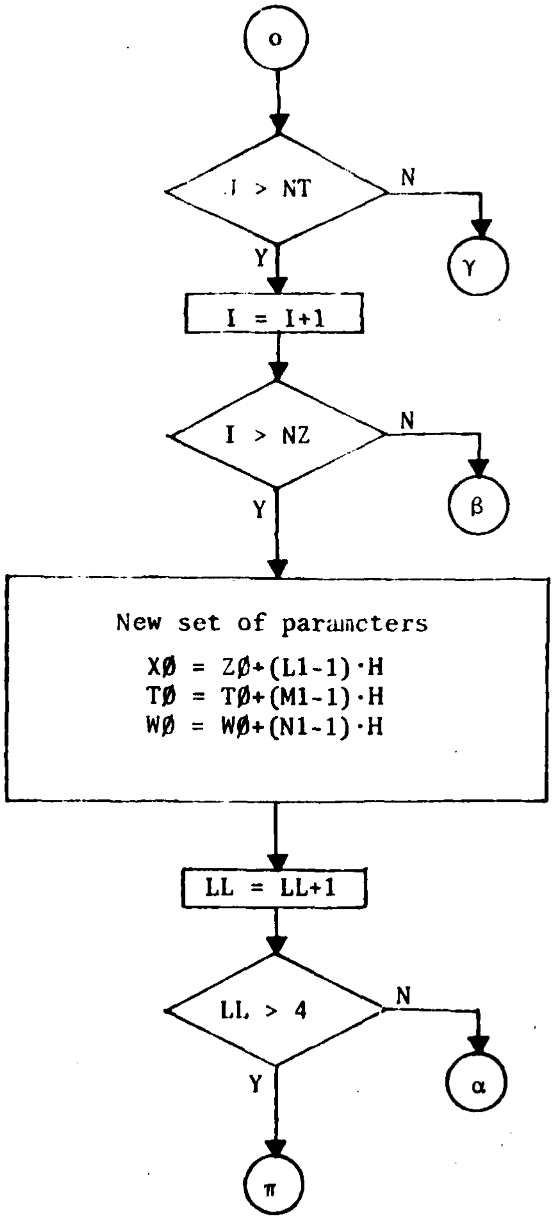
FP

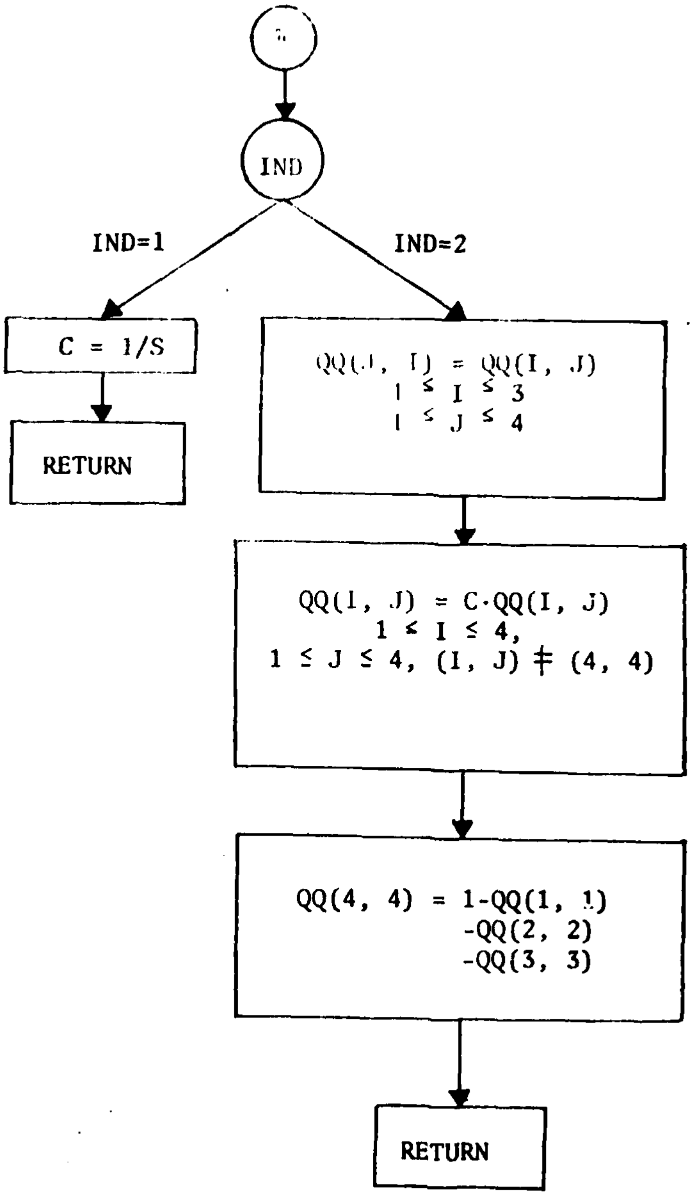
Flow Chart


BEGIN
PAI = PAI2/2
CØ = -50.0
H = HØ
Setting a parameter for scaling down
f(ϕ, θ, ψ)
F0 = FMAX-165.0
FØ ≤ 0.0
Y
N
FØ = 0.0
Preparation for local integration
ZØ = A(1)
TØ = A(2)
WØ = A(3)
L1 = (NZ/3) + 1
L2 = 4 · (NZ/6)
M1 = (NT/3) + 1
M2 = 4 · (NT/6)
N1 = (NW/3) + 1
N2 = 4 · (NW/6)
α
α
Location of mode
LØ = (NX/2) + 1
MØ = (NT/2) + 1
NØ = (NW/2) + 1
Initialization of
S, QQ, LL
S = 0.0, QQ(I,J) = 0.0
LL = 1
Volume Element
HD = H·H·H
I=1
β
I1=1
I<LØ
Y
N
I1 = I+1
X1 = ZØ+(I1-1)·H
J=1
σ


σ
γ
J1=J
J < MØ
Y
J1 = J+1
N
T1 = TØ+(J1-1)·H
T1 > PAI
Y
T1 = T1-PAI
N
T1 < 0.0
Y
T1 = T1+PAI
N
k=1
§
k1=k
k < NØ
Y
k1 = k+1
N
ρ
ρ
W1 = WØ+(k1-1)·H
LL=4
N
Y
L1 ≤ I ≤ L2
M1 ≤ J ≤ M2
N1 ≤ K ≤ N2
N
Y
ε
YY1 = f(φ1, θ1, ψ1)
(FP)
YY1 = YY1-FØ
YY1 ≤ CØ
Y
N
ε
YY = exp(YY1)
sin(T1)·H3
IND
IND=1
IND=2
S = S+YY
X1 = X1-W1
X2 = X1+W1
ε
ζ


ζ
W(1, 1) = (1-cos(T1))·(1+cos(X1))/4.0
W(2, 2) = (1-cos(T1))·(1-cos(X1))/4.0
W(3, 3) = (1+cos(T1))·(1-cos(X2))/4.0
W(4, 4) = (1+cos(T1))·(1+cos(X2))/4.0
W(1, 2) = (1-cos(T1))·sin(X1)/4.0
W(1, 3) = sin(T1)(sin(X1)+sin(W1))/4.0
W(1, 4) = sin(T1)·(cos(W1)+cos(X1))/4.0
W(2, 3) = sin(T1)·(cos(W1)-cos(X1))/4.0
W(2, 4) = sin(T1)·(sin(Z1)-sin(W1))/4.0
W(3, 4) = (1.0+cos(T1)) sin(X2)/4.0
QQ(I1, I2) = QQ(I1, I2)
+ YY·W(I1, I2)
1 ≤ I1 ≤ 3, I1 ≤ I2 ≤ 4
ε
k = k+1
k > NW
N
§
Y
J = J+1
o


o
J > NT
N
γ
Y
I = I+1
I > NZ
N
β
Y
New set of paramcters
XØ = ZØ+(L1-1)·H
TØ = TØ+(M1-1)·H
WØ = WØ+(N1-1)·H
LL = LL+1
LL > 4
N
α
Y
π


π
IND
IND=1
IND=2
C = 1/S
RETURN
QQ(J, I) = QQ(I, J)
1 ≤ I ≤ 3
1 ≤ J ≤ 4
QQ(I, J) = C·QQ(I, J)
1 ≤ I ≤ 4,
1 ≤ J ≤ 4, (I, J) ≠ (4, 4)
QQ(4, 4) = 1-QQ(1, 1)
-QQ(2, 2)
-QQ(3, 3)
RETURN

MAXI

A Subprogram

Purpose:

To locate the mode of a function $f(\phi, \theta, \psi)$ in the conditional density function.

$P(\phi, \theta, \psi) = C \exp (f(\phi, \theta, \psi))$, where C is a normalizing constant.

Calling Sequence:

CALL	MAXI (P, Q, QT, B, FMAX)

Common Variables:

PAI2

Input:

PAI2	2π
P, Q	$\{P^{\ell k}_{1mn}\}$, $\{P^{\ell k}_{2mn}\}$, 2x5x5-dim arrays
QT	Initial quaternion to start for searching, 4-dim array

Output:

B	Mode in terms of Euler angles, 3-dim array
FMAX	Value of the function $f(\phi, \theta, \psi)$ at the mode.

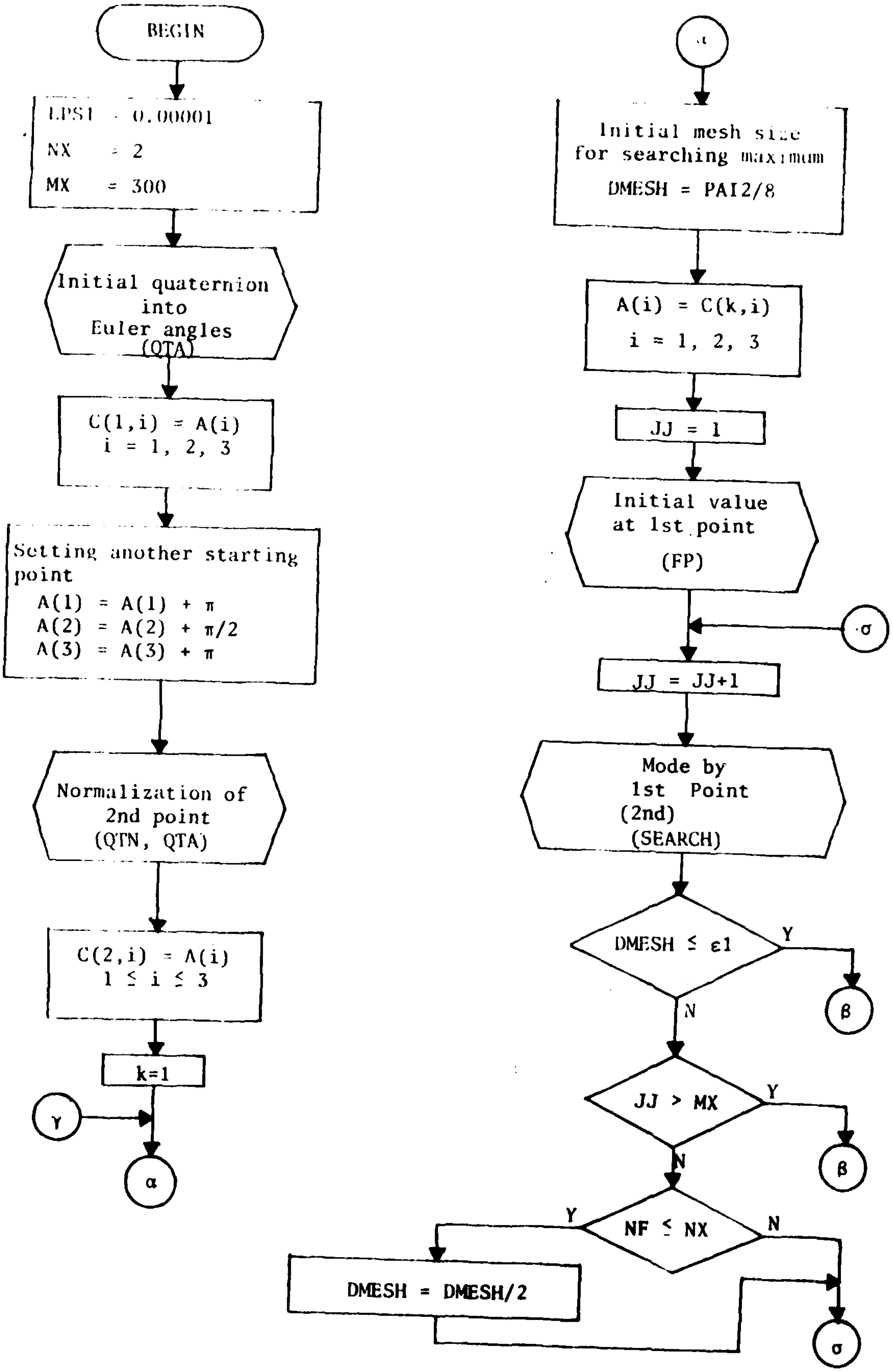
Special Considerations:

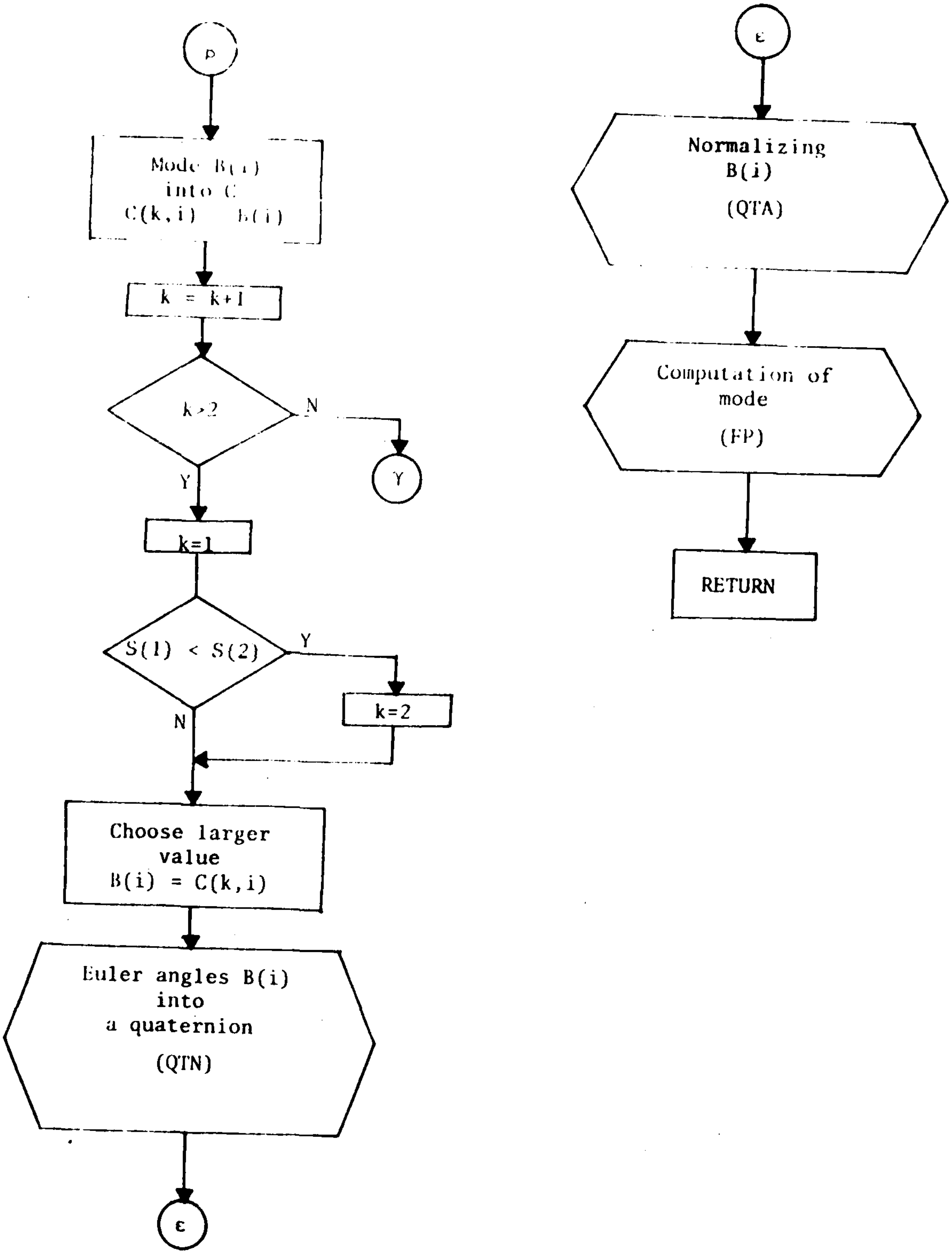
1. Two starting points are used to locate the mode. The first point (A(1), A(2), A(3)) is given by the calling program and the second point may be given by $(A(1)+\pi, A(2)+\pi/2, A(3)+\pi)$. This is to avoid locating a local maximum instead of the global maximum.

Other Subprograms Called:

QTA, QTN, FP, SEARCH

Flow Chart


BEGIN
EPS1 = 0.00001
NX = 2
MX = 300
Initial quaternion into Euler angles (QTA)
C(1,i) = A(i)
i = 1, 2, 3
Setting another starting point
A(1) = A(1) + π
A(2) = A(2) + π/2
A(3) = A(3) + π
Normalization of 2nd point (QTN, QTA)
C(2,i) = A(i)
1 ≤ i ≤ 3
k=1
γ
α
α
Initial mesh size for searching maximum
DMESH = PAI2/8
A(i) = C(k,i)
i = 1, 2, 3
JJ = 1
Initial value at 1st point (FP)
σ
JJ = JJ+1
Mode by 1st Point (2nd) (SEARCH)
DMESH ≤ ε1
Y
β
N
JJ > MX
Y
β
N
NF ≤ NX
Y
N
DMESH = DMESH/2
σ


P
Mode B(i)
into C
C(k,i) = B(i)
k = k+1
k>2
N
Y
Y
k=1
S(1) < S(2)
Y
k=2
N
Choose larger
value
B(i) = C(k,i)
Euler angles B(i)
into
a quaternion
(QTN)
ε
ε
Normalizing
B(i)
(QTA)
Computation of
mode
(FP)
RETURN

MØ6PLT

Main program

Purpose:

To read processes $\{s_k\}$, $\{\hat{s}_k\}$, $\{\hat{\hat{s}}_k\}$ and sequences of error estimates from the disk and draw graphs.

Common Variables:

None.

Input:

A   1742-dim. array containing all the values.

Special Considerations:

1. Processes s_k, $\hat{s}_k$, $\hat{\hat{s}}_k$ are called QSN, QH1, QH2 in MØ5S03 and stored in the location.

   A(568) ↔ QSN(1,1)

   A(768) ↔ QH1(1,1)

   A(968) ↔ QH2(1,1)

2. Optimal estimate errors, $\rho(s_k, \hat{s}_k)$, and $\rho(s_k, \hat{\hat{s}}_k)$, $\rho(\hat{s}_k, \hat{\hat{s}}_k)$ are called ERRN, ERH1, ERH2, ER12 in MØ5S03 and stored in the location

   A(268) ↔ ERRN(1)

   A(318) ↔ ERH1(1)

   A(368) ↔ ERH2(1)

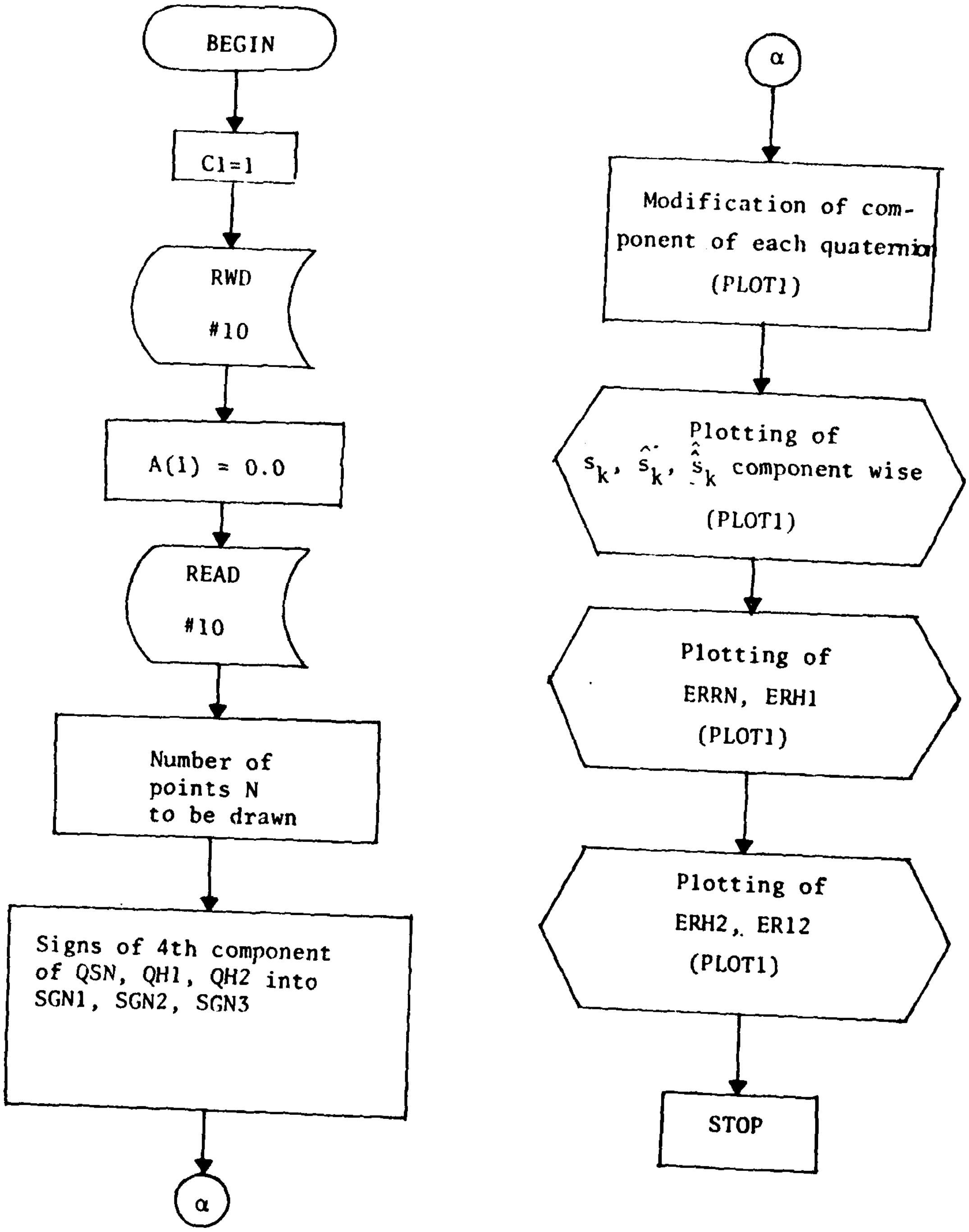
   A(418)   ER12(1)

3. Quaternion representation QSN, QH1, QH2 of s_k, $\hat{s}_k$, $\hat{\hat{s}}_k$ are modified so that 4th component of each quarternion is positive.

Other subprograms called.

PLOTI.

Flow Chart


BEGIN
C1=1
RWD
#10
A(1) = 0.0
READ
#10
Number of
points N
to be drawn
Signs of 4th component
of QSN, QH1, QH2 into
SGN1, SGN2, SGN3
α
α
Modification of com-
ponent of each quaternion
(PLOT1)
Plotting of
s_k, $\hat{s}_k$, $\hat{\hat{s}}_k$ component wise
(PLOT1)
Plotting of
ERRN, ERH1
(PLOT1)
Plotting of
ERH2, ER12
(PLOT1)
STOP

NEWØ

A subprogram

Purpose:

To generate $w_k(\phi,\theta,\psi)$ in the process.

$$s_{k+1} = w_k \circ s_k$$

using a pseudorandom number generator.

Calling Sequence:

CALL NEWϕ (IXϕ, PAI2, Wϕ)

Common Variables:

None

Input:

IXO	initial random integer
PAI2	2π

Output

WØ	w_k, 3 dim. array representing Euler's angles ϕ,θ,ψ.

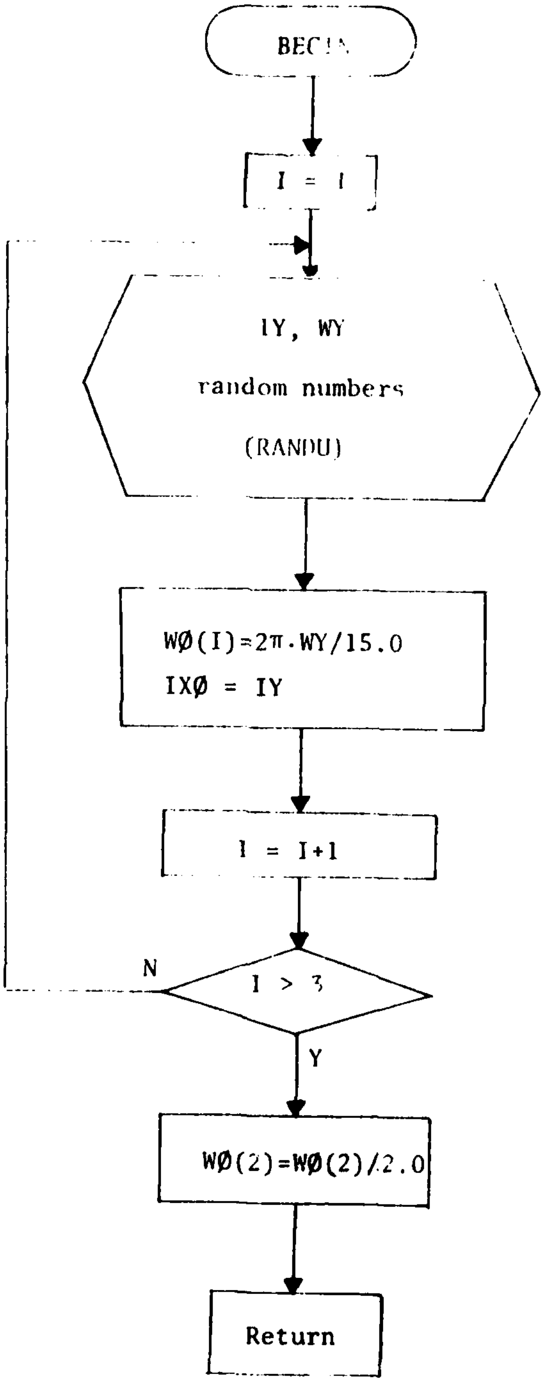
Special Considerations

1. Only three numbers are chosen from the start by RANDU.

Other Subprogram Called:

RANDU.

Flow Chart


BEGIN
I = 1
IY, WY
random numbers
(RANDU)
WØ(I)=2π·WY/15.0
IXØ = IY
I = I+1
N
I > 3
Y
WØ(2)=WØ(2)/2.0
Return

NPAQ

A subprogram

Purpose.

To update $\{P^{\ell k-1}_{1mn}\}$ and $\{P^{\ell k-1}_{2mn}\}$ of $P(s_{k-1}|m^{k-1})$ to $\{P^{\ell k}_{1mn}\}$ and $\{P^{\ell k}_{2mn}\}$ of $P(s_k|m^k)$, where

$$P(s_k|m^k) = C \exp \sum_{\ell=1}^{2} \frac{1}{2}P^{\ell k}_{100}D_{00}(s_k) + \sum_{\substack{m,n=-\ell \\ m<n \\ \text{or} \\ m=n>0}}^{\ell} (P^{\ell k}_{1mn}\mathrm{Re}D^{\ell}_{mn}(s_k)$$

$$+ P^{\ell k}_{2mn}\mathrm{Im}D^{\ell}_{mn}(s_k))$$

Calling Sequence:

CALL NPAQ(PR)

Common Variables

AA, BB, WØ, OBN, PØ, QØ, P1 Q1

Input

RR	constants given in the main program.
AA, BB	coefficients of the functions $(-\underline{u}_1, \underline{u}_3, -\underline{\underline{u}}_1, \underline{\underline{u}}_3, \underline{u}_2, \underline{\underline{u}}_2)$.
OBN	observed value computed in the main program 6-dim. array, $\{m^j_k\}$
PØ	$\{P^{\ell k-1}_{1mn}\}$, 2x5x5 dim. array
QØ	$\{P^{\ell k-1}_{2mn}\}$, 2x5x5 dim. array

Output:

P1	$\{P^{\ell k}_{1mn}\}$
Q1	$\{P^{\ell k}_{2mn}\}$

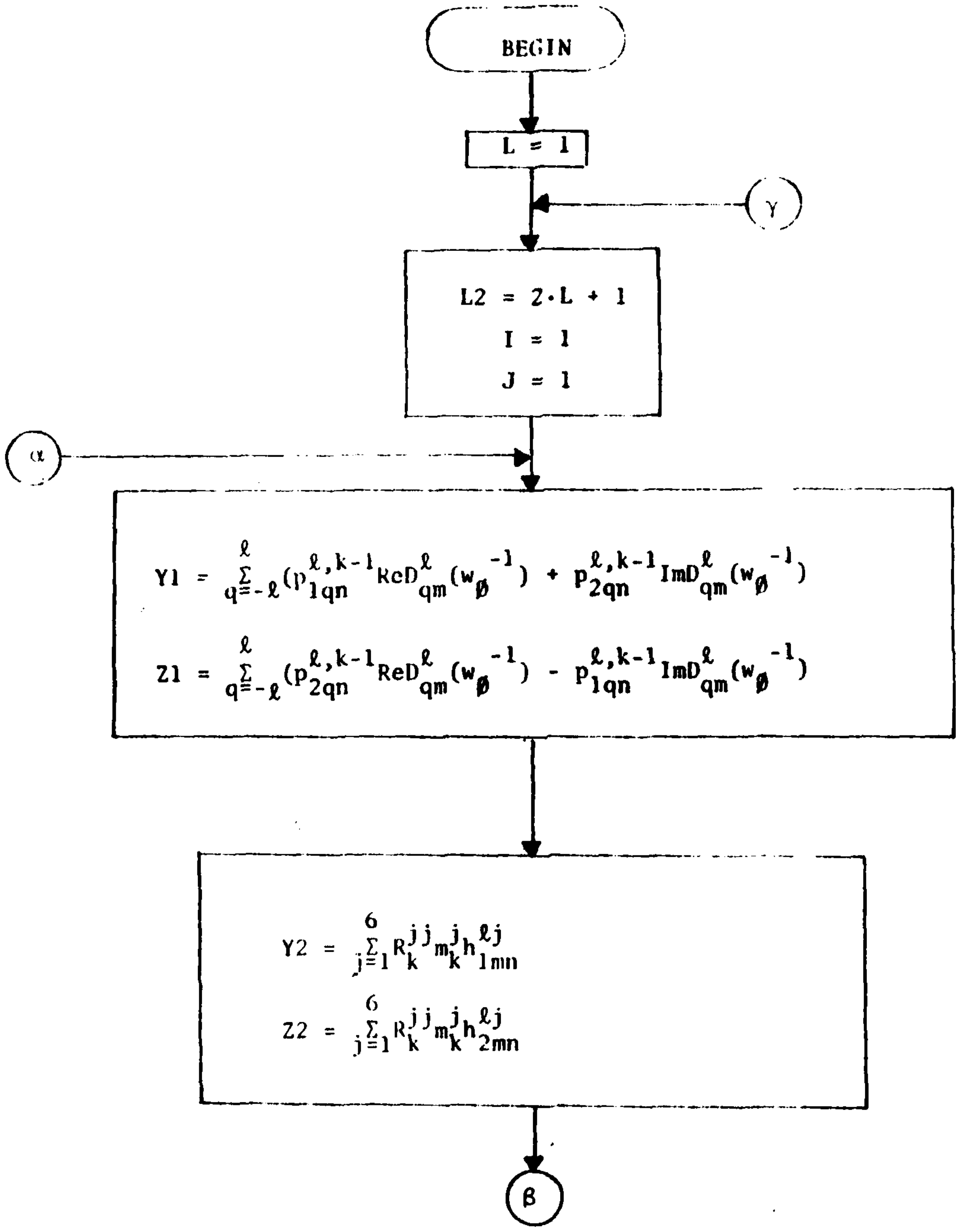
Special Considerations:

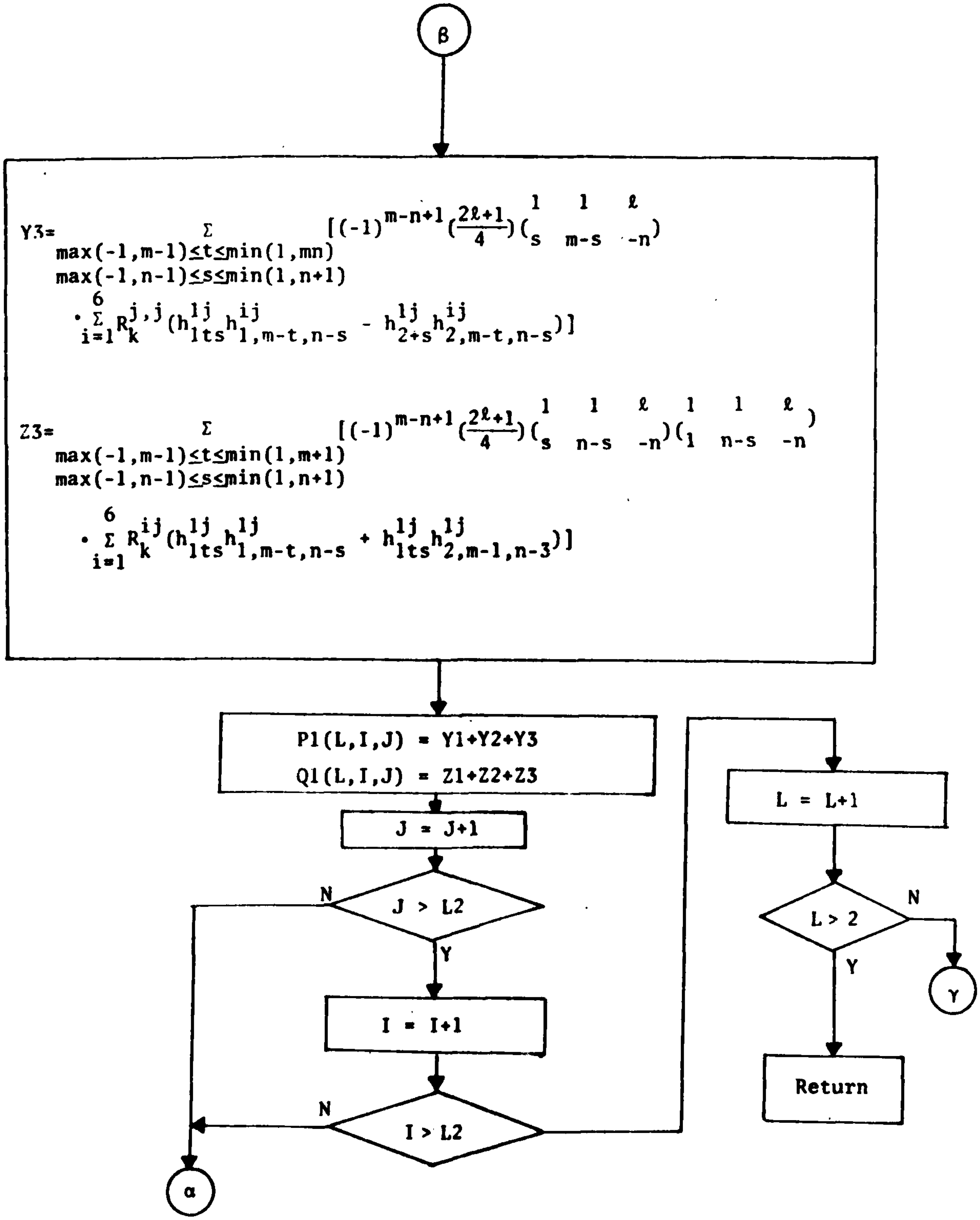
None.

Other Subprogram Called:

SD, DRI1, C3J

Flow Chart


BEGIN
L = 1
γ
L2 = 2·L + 1
I = 1
J = 1
α
$Y1 = \sum_{q=-\ell}^{\ell}(p_{1qn}^{\ell,k-1} ReD_{qm}^{\ell}(w_{\phi}^{-1}) + p_{2qn}^{\ell,k-1} ImD_{qm}^{\ell}(w_{\phi}^{-1})$
$Z1 = \sum_{q=-\ell}^{\ell}(p_{2qn}^{\ell,k-1} ReD_{qm}^{\ell}(w_{\phi}^{-1}) - p_{1qn}^{\ell,k-1} ImD_{qm}^{\ell}(w_{\phi}^{-1})$
$Y2 = \sum_{j=1}^{6} R_k^{jj} m_k^j h_{1mn}^{\ell j}$
$Z2 = \sum_{j=1}^{6} R_k^{jj} m_k^j h_{2mn}^{\ell j}$
β



$$Y3=\sum_{\substack{\max(-1,m-1)\le t\le\min(1,mn)\\ \max(-1,n-1)\le s\le\min(1,n+1)}}\Big[(-1)^{m-n+1}\Big(\frac{2\ell+1}{4}\Big)\begin{pmatrix}1 & 1 & \ell\\ s & m-s & -n\end{pmatrix}\cdot\sum_{i=1}^{6}R_k^{j,j}\big(h^{1j}_{1ts}h^{ij}_{1,m-t,n-s}-h^{1j}_{2+s}h^{ij}_{2,m-t,n-s}\big)\Big]$$

$$Z3=\sum_{\substack{\max(-1,m-1)\le t\le\min(1,m+1)\\ \max(-1,n-1)\le s\le\min(1,n+1)}}\Big[(-1)^{m-n+1}\Big(\frac{2\ell+1}{4}\Big)\begin{pmatrix}1 & 1 & \ell\\ s & n-s & -n\end{pmatrix}\begin{pmatrix}1 & 1 & \ell\\ 1 & n-s & -n\end{pmatrix}\cdot\sum_{i=1}^{6}R_k^{ij}\big(h^{1j}_{1ts}h^{1j}_{1,m-t,n-s}+h^{1j}_{1ts}h^{1j}_{2,m-1,n-3}\big)\Big]$$

PLOT1

A subprogram

Purpose:

To draw one, two or three graphs with different type of lines.

Calling Sequence:

```
CALL     PLOT1(M, N, X, Y, Z, W, U)
```

Common Variables:

None.

Input:

M	number of graphs to be drawn
N	number of points to be drawn
Y,Z,W	values for graphs
U	a dummy array

Special Consideration

1. X-coordinates are determined by

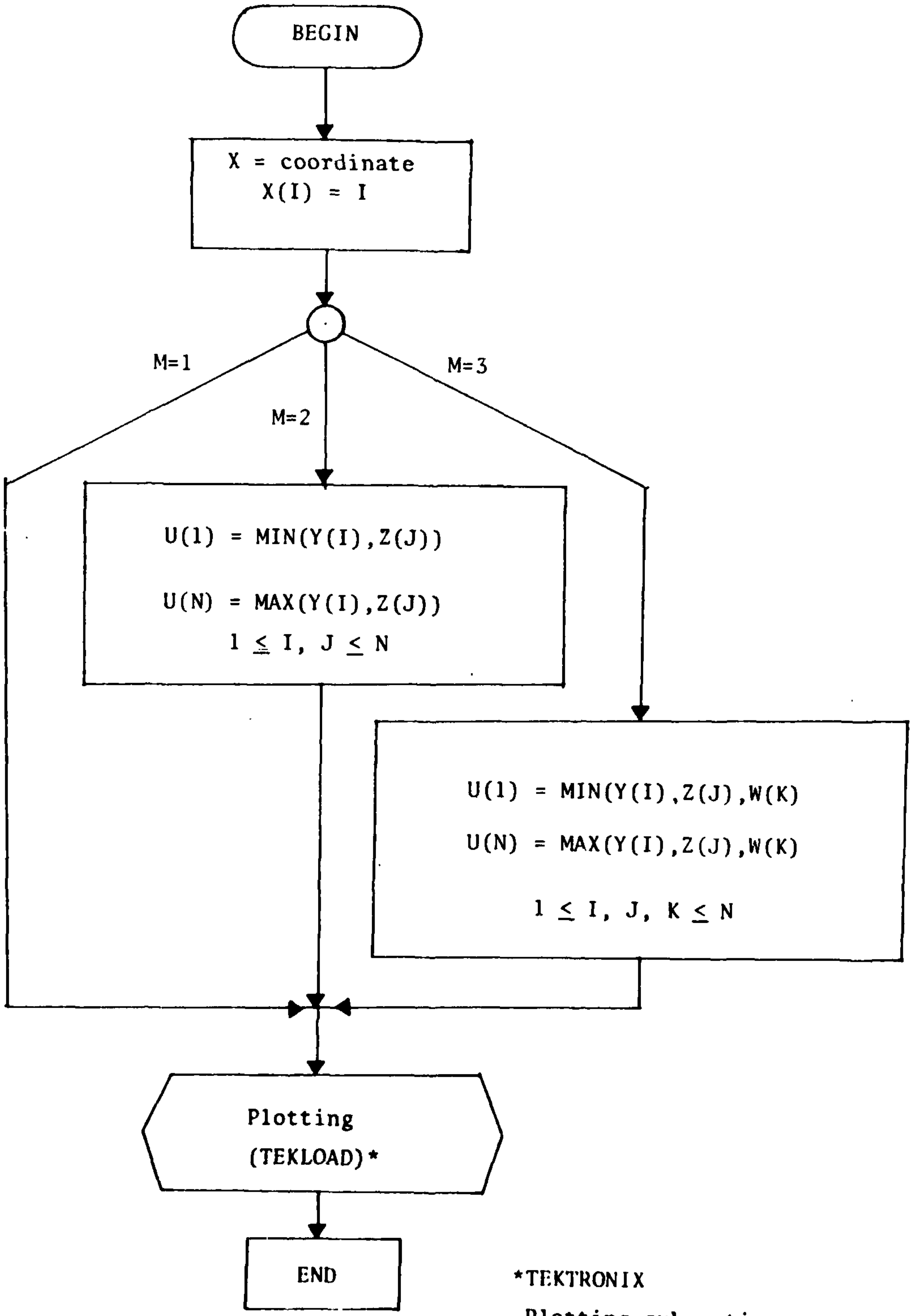
   X(I) = I

2. More than one graph in one picture are drawn completely checking the size of y-values using UPDWN.

Other Subprograms Called:

UPDWN, INITT, BINITT, NPTS, CHECK, DSPLAY, LINE, CPLOT, TINPUT.

Flow Chart



*TEKTRONIX
Plotting subroutines

QTA

A Subprogram

Purpose:

To convert a quaternion to Euler angles.

Calling Sequence:

CALL QTA (Q, A)

Common Variables:

None

Input:

Q	Quaternion, 4-dim array

Output:

A	Euler angles, 3-dim array

Special Considerations:

1. The following relations between a quaternion and Euler angles are used.

$$q_1 = \sin\frac{\theta}{2} \cos\ ((\phi-\psi)/2)$$

$$q_2 = \sin\frac{\theta}{2} \sin\ ((\phi-\psi)/2)$$

$$q_3 = \cos\frac{\theta}{2} \sin\ ((\phi+\psi)/2)$$

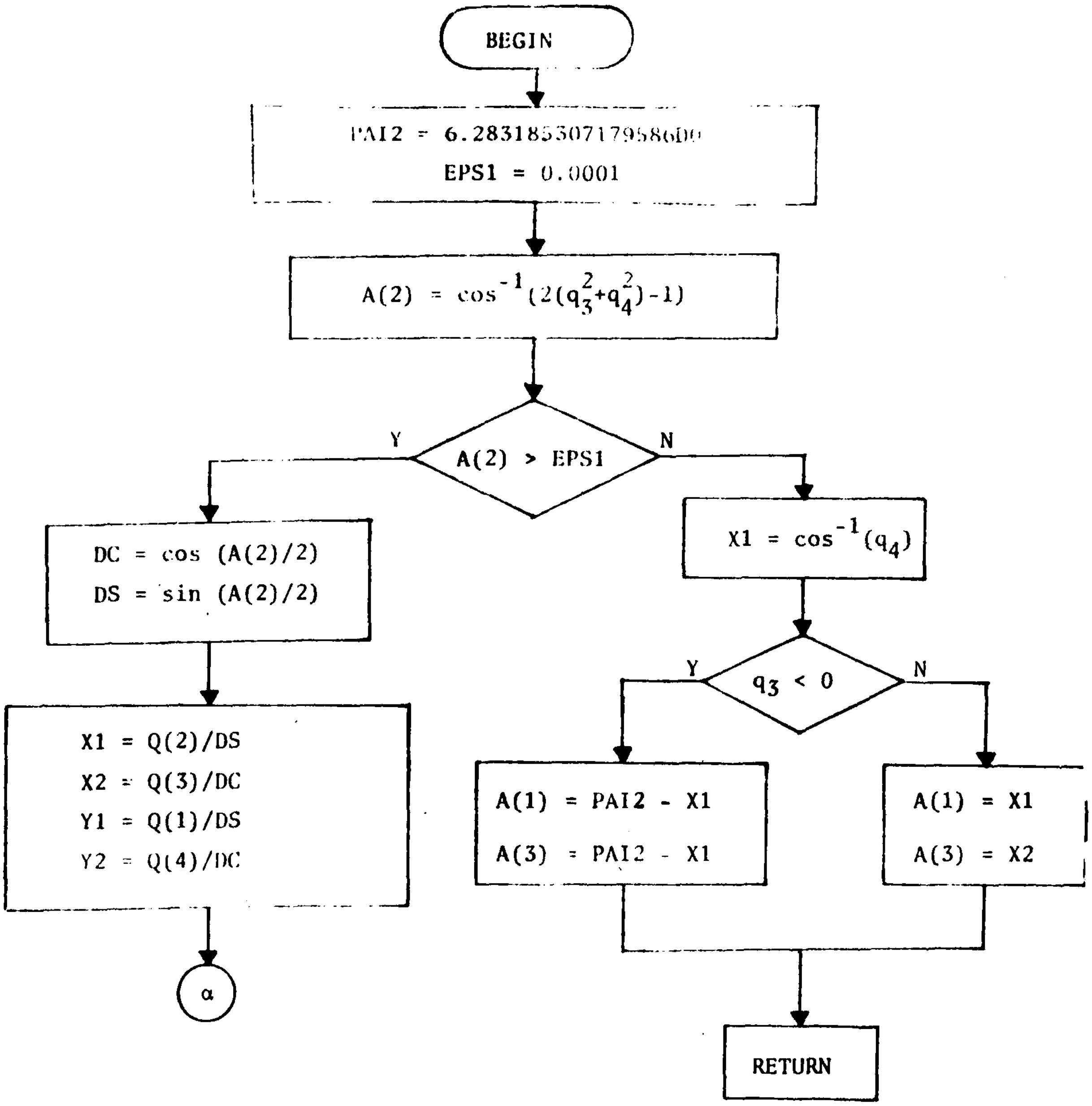
$$q_4 = \cos\frac{\theta}{2} \cos\ ((\phi+\psi)/2)$$

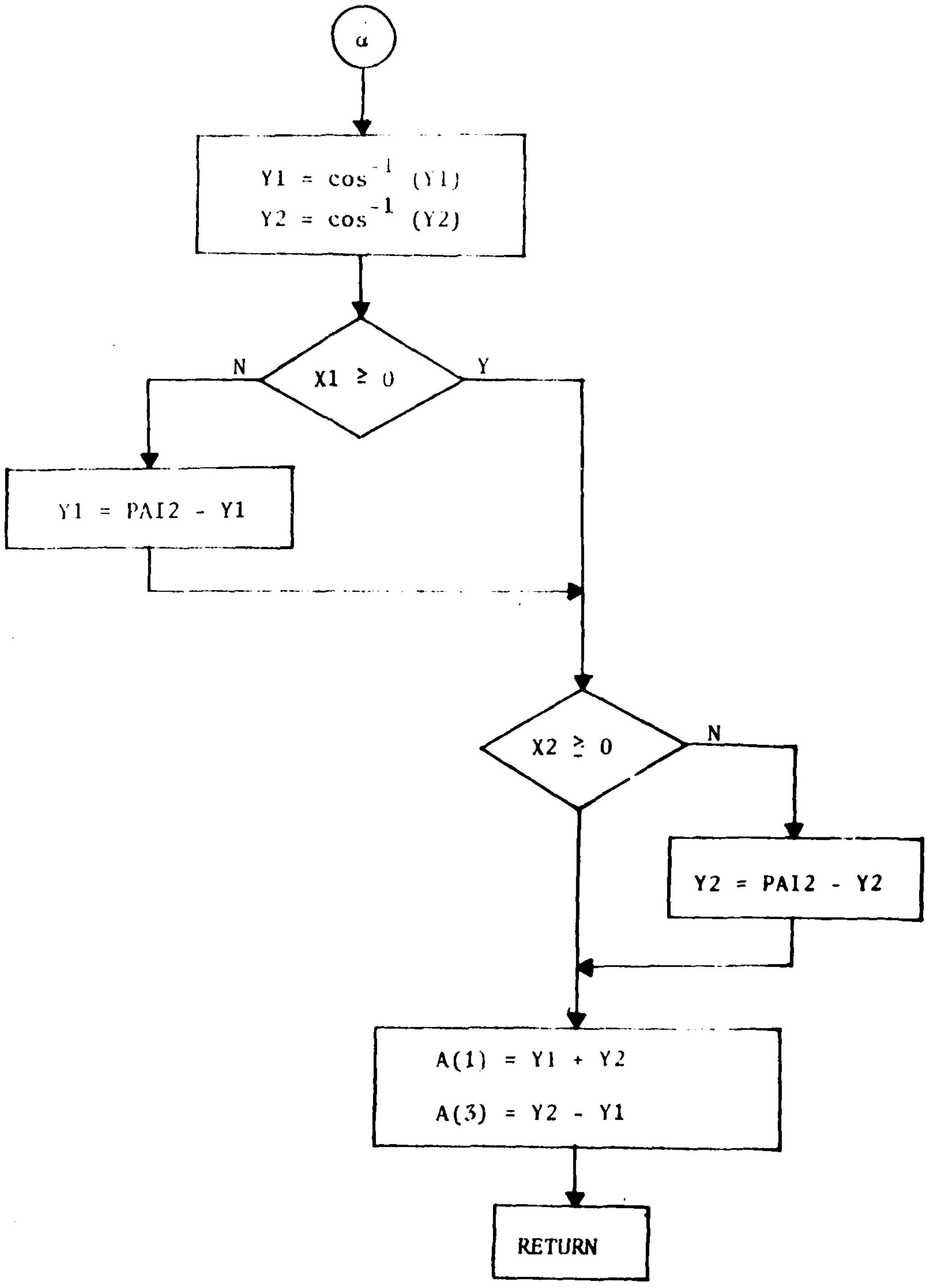
2. If Arcos($2(q_3^2 + q_4^2) - 1) < \mathrm{EPS1}$, then θ is assumed to be zero.

Other Subprograms Called:

None.

Flow Chart


BEGIN
PAI2 = 6.283185307179586D0
EPS1 = 0.0001
A(2) = cos^{-1}(2(q_3^2+q_4^2)-1)
A(2) > EPS1
Y
N
DC = cos (A(2)/2)
DS = sin (A(2)/2)
X1 = Q(2)/DS
X2 = Q(3)/DC
Y1 = Q(1)/DS
Y2 = Q(4)/DC
α
X1 = cos^{-1}(q_4)
q_3 < 0
Y
N
A(1) = PAI2 - X1
A(3) = PAI2 - X1
A(1) = X1
A(3) = X2
RETURN


a
$Y1 = \cos^{-1}(Y1)$
$Y2 = \cos^{-1}(Y2)$
N
X1 ≥ 0
Y
Y1 = PAI2 - Y1
X2 ≥ 0
N
Y2 = PAI2 - Y2
A(1) = Y1 + Y2
A(3) = Y2 - Y1
RETURN

QTN

A Subprogram

Purpose

To transform Euler angles to a quaternion

Calling Sequence

CALL QTN(A, Q)

Common Variables

None.

Input

A Euler angles (ϕ,θ,ψ), 3 dim. array.

Output

Q Quaternion (q_1, q_2, q_3, q_4), 4-dim. array.

Special Considerations

1. The formula for the transformation is given by

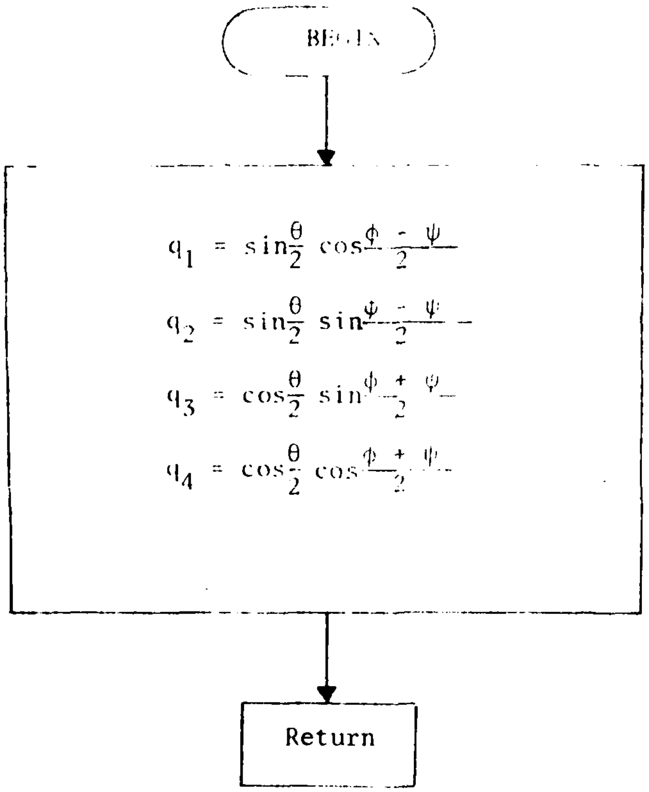
$$q_1 = \sin\frac{\theta}{2}\ \cos\ \frac{\phi-\psi}{2}$$

$$q_2 = \sin\frac{\theta}{2}\ \sin\ \frac{\phi-\psi}{2}$$

$$q_3 = \cos\frac{\theta}{2}\ \sin\ \frac{\phi+\psi}{2}$$

$$q_4 = \cos\frac{\theta}{2}\ \cos\ \frac{\phi+\psi}{2}$$

Flow Chart


BEGIN
$q_1 = \sin\frac{\theta}{2} \cos\frac{\phi - \psi}{2}$
$q_2 = \sin\frac{\theta}{2} \sin\frac{\phi - \psi}{2}$
$q_3 = \cos\frac{\theta}{2} \sin\frac{\phi + \psi}{2}$
$q_4 = \cos\frac{\theta}{2} \cos\frac{\phi + \psi}{2}$
Return

A subprogram

Purpose:

To compute the distance between two quatenions.

Calling Sequence:

```
CALL QVQ(D, Q, P)
```

Common Variables:

None

Input:

P; Q	quaternions, 4-dim. arrays.

Output:

D	distance between P and Q.

Special Consideration:

1. Distance between two quatenions p and q is given by

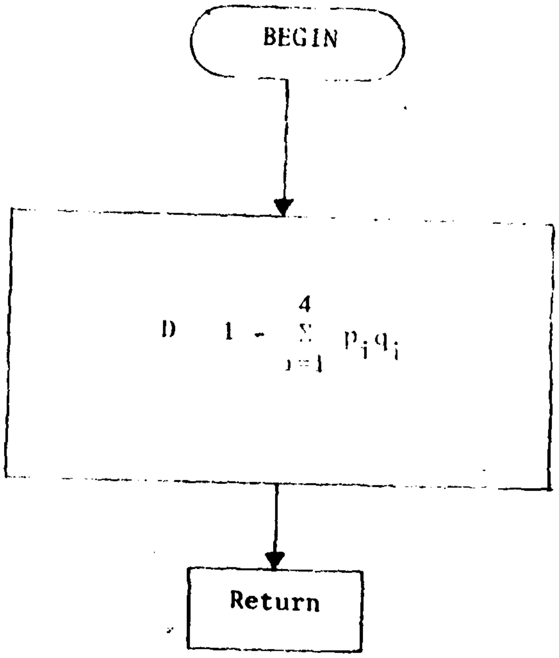
$$d = 1 - \sum_{i=1}^{4} p_i \cdot q_i , \text{ where } \quad p = (p_1, p_2, p_3, p_4)$$

$$q = (q_1, q_2, q_3, q_4)$$

Other Subprograms Called:

None.

Flow Chart



QXQ

A subprogram

Purpose:

To compute the product of two quaternions

$(r_1, r_2, r_3, r_4) = (P_1, P_2, P_3, P_4)\cdot(q_1, q_2, q_3, q_4)$,

where

$$r_1 = P_1q_4 + P_2q_3 - P_3q_2 + P_4q_1$$

$$r_2 = P_1q_3 + P_2q_4 + P_3q_1 + P_4q_2$$

$$r_3 = P_1q_2 - P_2q_1 + P_3q_4 + P_4q_3$$

$$r_4 = -P_1q_1 - P_2q_2 - P_3q_3 + P_4q_4$$

Calling Sequence:

CALL QXQ(P, Q, R)

Common Variables:

None.

Input:

P	quarternion (P_1, P_2, P_3, P_4), 4-dim. array.
Q	quarternion (q_1, q_2, q_3, q_4), 4-dim. array.

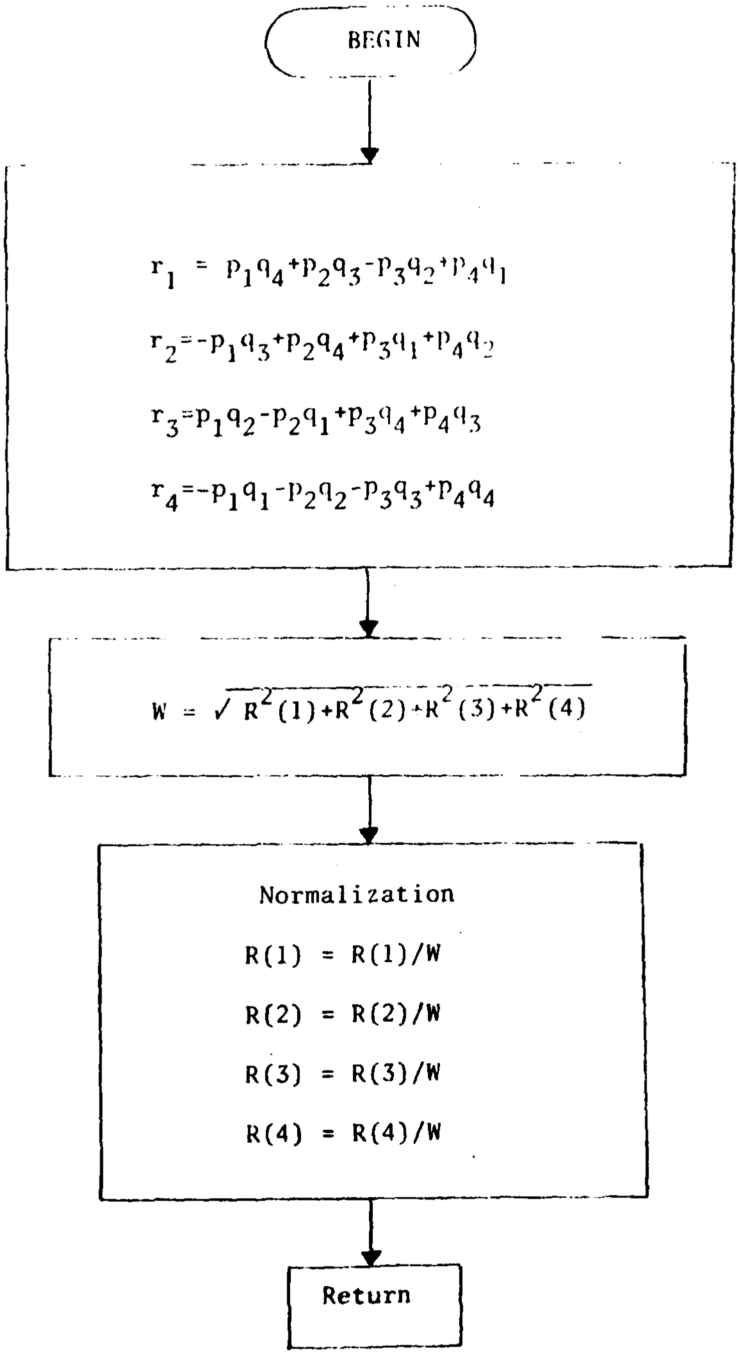
Output:

R	quarternion (r_1, r_2, r_3, r_4), 4-dim. array.

Special considerations

1. Considering computer round-off error, product quarternion R is re-normalized.

Flow Chart


BEGIN
$r_1 = p_1q_4+p_2q_3-p_3q_2+p_4q_1$
$r_2=-p_1q_3+p_2q_4+p_3q_1+p_4q_2$
$r_3=p_1q_2-p_2q_1+p_3q_4+p_4q_3$
$r_4=-p_1q_1-p_2q_2-p_3q_3+p_4q_4$
$W = \sqrt{R^2(1)+R^2(2)+R^2(3)+R^2(4)}$
Normalization
R(1) = R(1)/W
R(2) = R(2)/W
R(3) = R(3)/W
R(4) = R(4)/W
Return

RANDU

A subprogram

Purpose:

To compute a pseudorandom integer and value of uniform distribution.

The program uses the properties of FORTRAN IV and its admissible maximum integer $2^{31}-1$ (=2147483647) and overflows, (the program is taken from IBM 360 scientific subroutine,p.77).

Calling Sequence:

CALL RANDU(IX, IY, YFL)

Common Variables:

None.

Input:

IX	integer

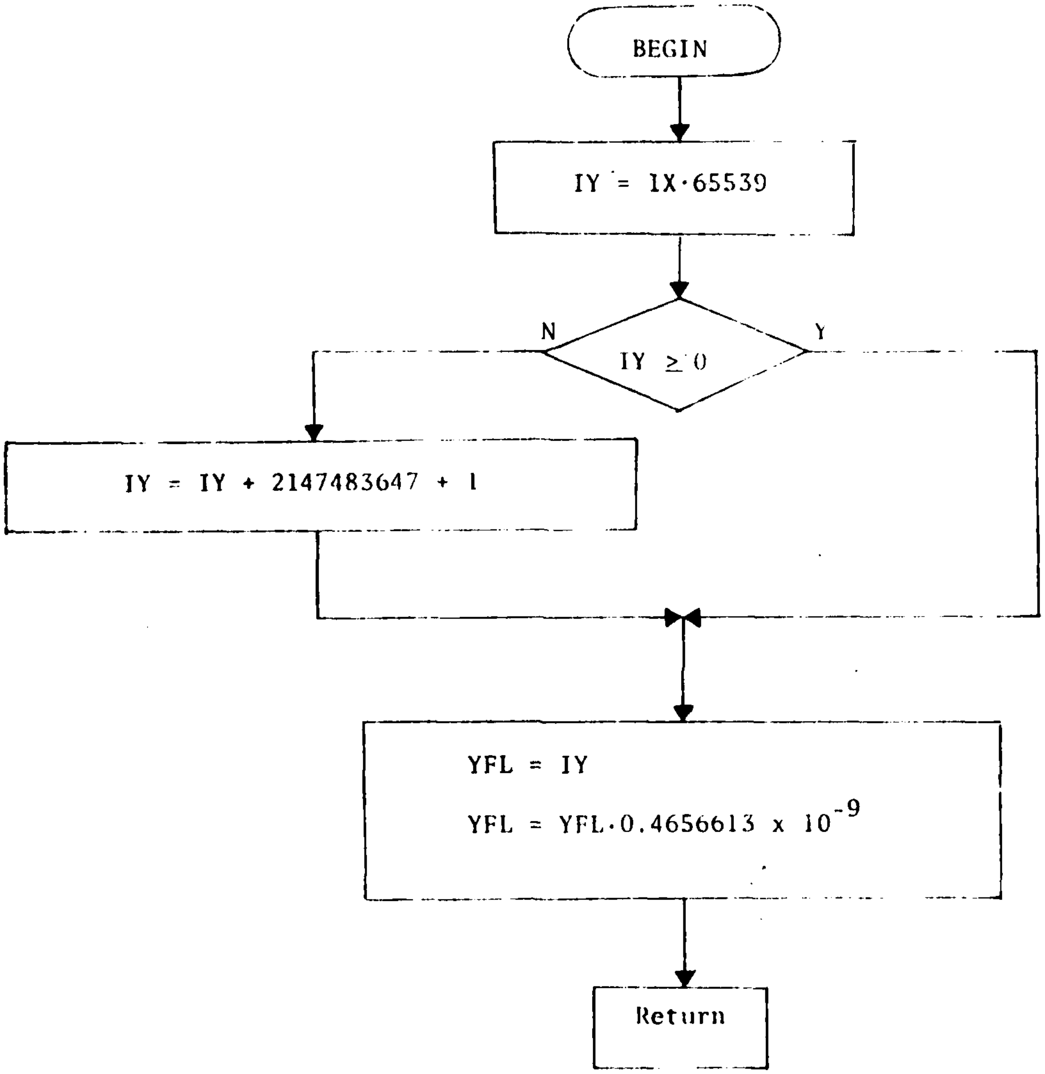
Output:

IY	pseudorandom integer
YFL	pseudorandom value, YFL = $IY/(2^{31}-1)$

Special Considerations:

1. To generate a sequence of pseudorandom numbers RANDU can be used repeatedly by setting IX = IY.

Flow Chart


BEGIN
IY = IX·65539
IY ≥ 0
N
Y
IY = IY + 2147483647 + 1
YFL = IY
YFL = YFL·0.4656613 x 10^{-9}
Return

SINCOS

A subprogram

Purpose

To construct numerical tables of sinx, cosx for $0 \leq x \leq 2\pi$

Calling Sequence:

```
CALL     SINCOS(N, SINE, COSN)
```

Common Variables:

None

Input:

N   Number of points in $[0, 2\pi]$ to be taken

Output:

SINE   values $\{\sin x_i\}$, N-dim. array

COSN   values $\{\cos x_i\}$, N-dim. array

Special Considerations:

1. Tables are used to reduce computing time.
2. In order to avoid future underflows,

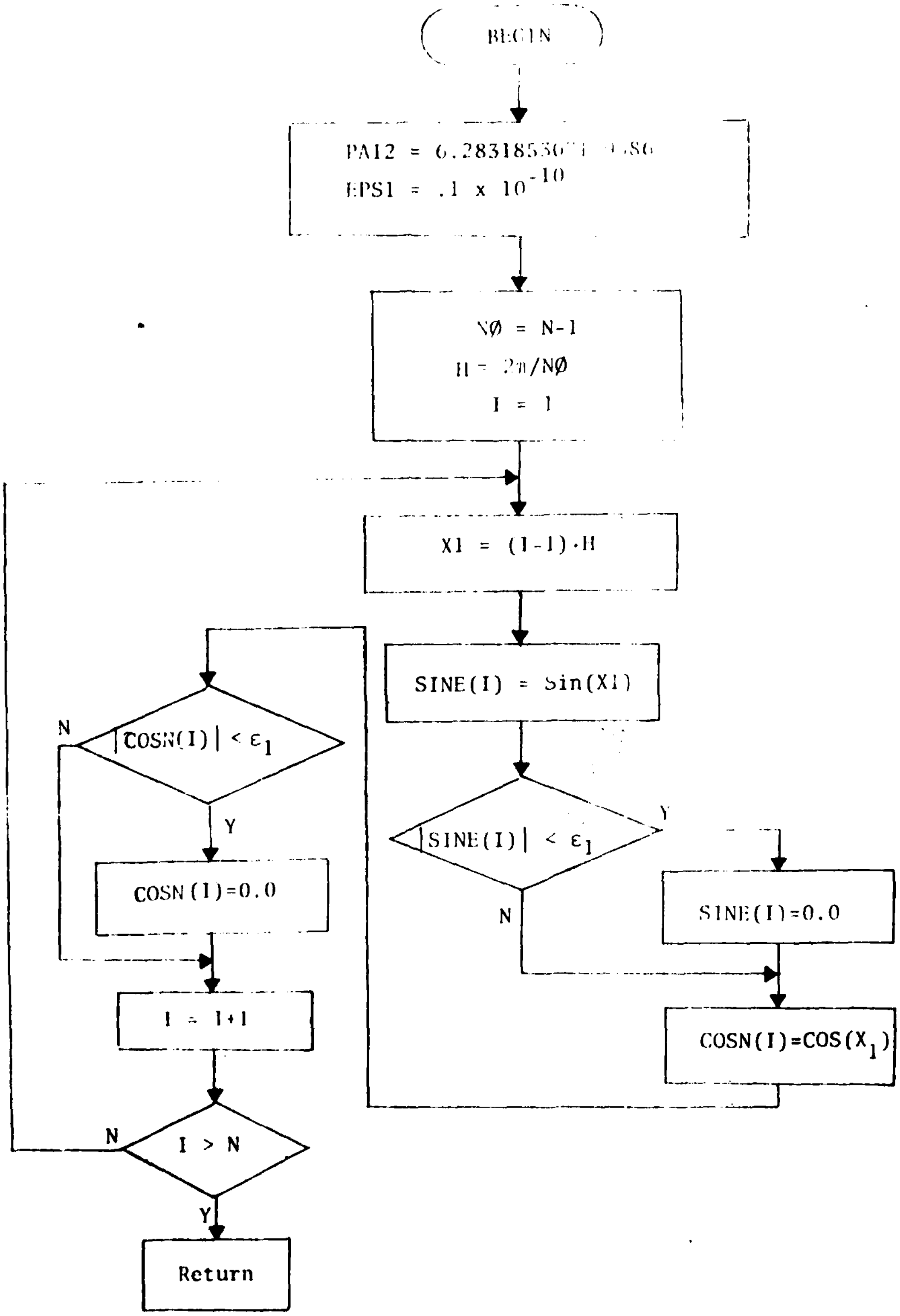
   if $|SINE(I)| \quad (|COS\ N(I)|) < 10^{-11}$, then

   $SINE(I) = 0 \quad (COS\ N(I)=0)$

Other Subprograms Called:

None.

Flow Chart


BEGIN
PAI2 = 6.283185307179586
EPS1 = .1 x 10^-10
NØ = N-1
H = 2π/NØ
I = 1
X1 = (I-1)·H
SINE(I) = Sin(X1)
|SINE(I)| < ε1
Y
N
SINE(I)=0.0
COSN(I)=COS(X1)
|COSN(I)| < ε1
N
Y
COSN(I)=0.0
I = I+1
I > N
N
Y
Return

SD

A subprogram

Purpose:

To assign values $d^{\ell}_{mn}(T)$ to each ℓ,m,n and $0 \le T \le \pi$

Calling Sequence:

Y = SD(L,M,N,T)

Input:

L order of Fourier coefficents to be computed

M,N indices of Fourier coefficients with the range $-L \le M,N \le L$

T angle $0 \le T \le \pi$

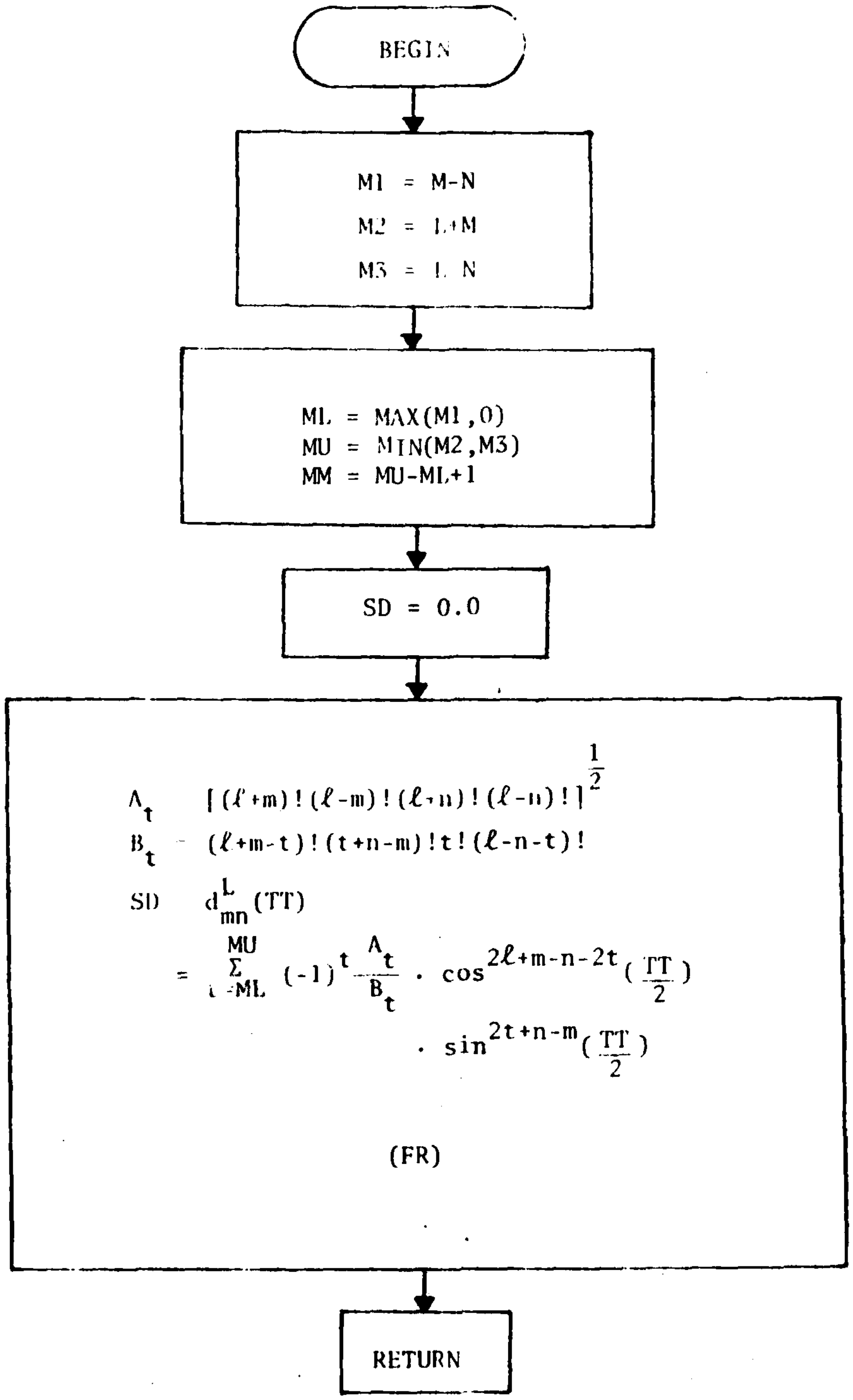
Special Considerations:

1. L,M,N must satisfy the relations

$0 \le L \le$ LMAX and $-L \le M,N \le L$

Other Subprograms Called:

FR


BEGIN
M1 = M-N
M2 = L+M
M3 = L-N
ML = MAX(M1,0)
MU = MIN(M2,M3)
MM = MU-ML+1
SD = 0.0
A_t $[(\ell+m)!(\ell-m)!(\ell+n)!(\ell-n)!]^{\frac{1}{2}}$
B_t - $(\ell+m-t)!(t+n-m)!t!(\ell-n-t)!$
SD $d^{L}_{mn}(TT)$
$= \sum_{t=ML}^{MU} (-1)^t \frac{A_t}{B_t} \cdot \cos^{2\ell+m-n-2t}(\frac{TT}{2})$
$\cdot \sin^{2t+n-m}(\frac{TT}{2})$
(FR)
RETURN

SEARCH

A subprogram

Purpose:

To find the mode of the conditional density function with a given initial starting point and mesh size.

Calling sequence:

CALL SEARCH (DMESH, NF, P, Q, A, B, S1)

Common Variables:

PAI2

Input:

PAI2 2π

DMESH Stepsize to search the maximum value of the density function.

NF Maximum searching times in one direction at the previous stage.

P, Q $\{P^{\ell k}_{1mn}\}$, $\{P^{\ell k}_{2mn}\}$, 2x5x5-dim arrays

A Starting point (Euler angles), 3-dim array

S1 Value of $f(\phi, \theta, \psi)$ at A

Output:

B Maximum point (mode) of the function $f(\phi, \theta, \psi)$ with respect to the given mesh

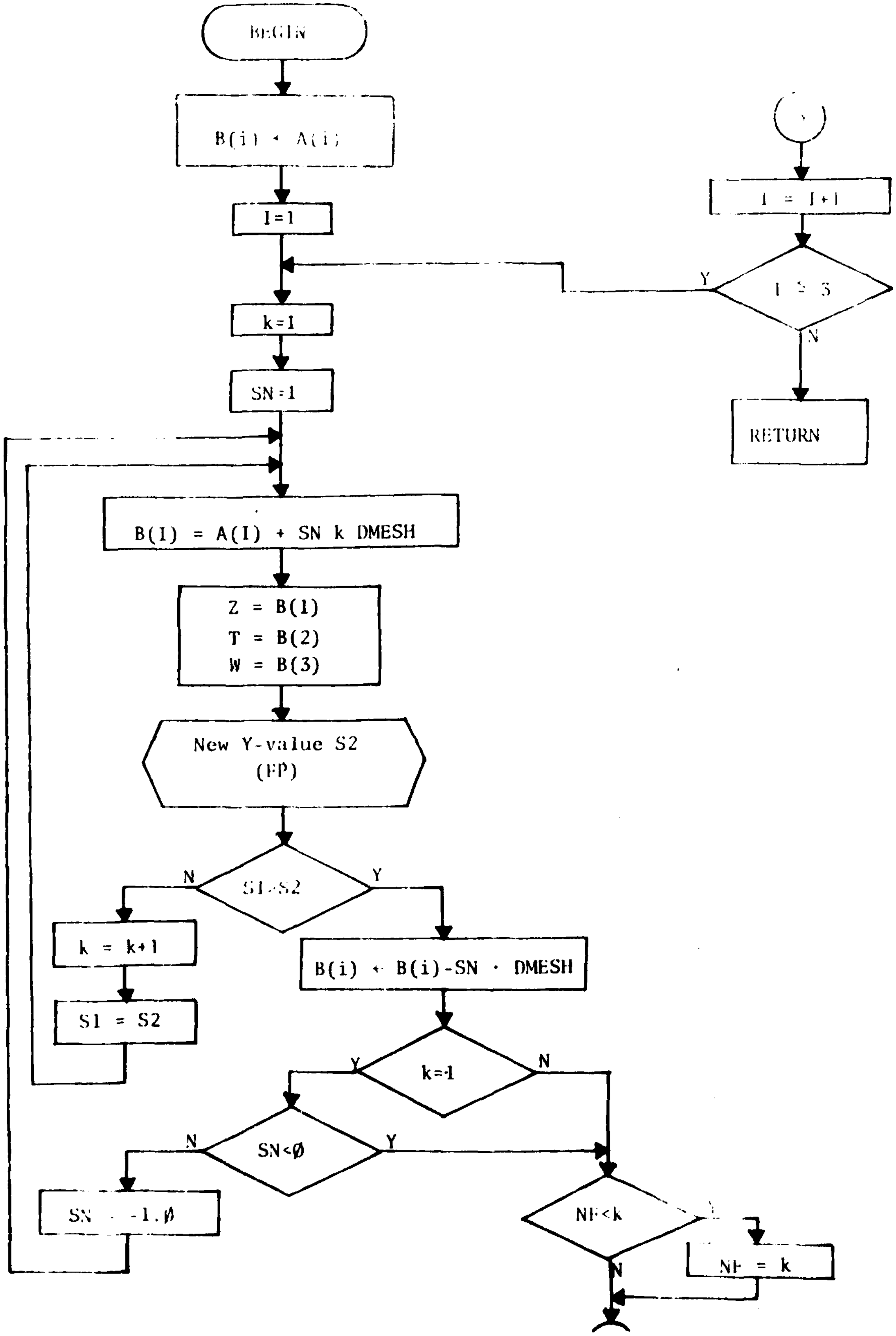
S1 Maximum value of $f(\phi, \theta, \psi)$

NF Maximum searching times at this stage, which will be used in calling program 'MAXI' to determine the new mesh size

Special Considerations:

1. Searching the mode continues as long as the previous function value is less than the new value.

Flow Chart


BEGIN
B(i) ← A(i)
I=1
k=1
SN=1
B(I) = A(I) + SN k DMESH
Z = B(1)
T = B(2)
W = B(3)
New Y-value S2
(FP)
S1<S2
N
Y
k = k+1
S1 = S2
B(i) ← B(i)-SN · DMESH
k=1
Y
N
SN<Ø
N
Y
SN ← -1.Ø
NF<k
Y
N
NF = k
I = I+1
I ≤ 3
Y
N
RETURN

A subprogram

Purpose:

1. To store processes

   $\{s_i\}$ $\{\hat{s}_i\}$ $\{\hat{\hat{s}}_i\}$ $1 \leq i \leq (k-1)$, where $\{s_i\}$ are attitudes, $\{\hat{s}_i\}$ attitude estimates, $\{\hat{\hat{s}}_i\}$ maximum likelihood estimates, and

   $$\mathrm{ERRN}(i) = E(\rho(s_i, \hat{s}_i)),\ \mathrm{ERH1}(i) = \rho(s_i, \hat{s}_i),$$

   $$\mathrm{ERH2}(i) = \rho(s_i, \hat{\hat{s}}_i) \qquad \mathrm{ER12}(i) = \rho(\hat{s}_i, \hat{\hat{s}}_i).$$

2. To save the values of the parameters for the next step k.

3. To read all the results from disk and printout if requested.

Calling Sequence:

CALL TRANS(ID, IXØ, IYØ, KK, MM)

Common Variables:

AXX, AD, AA, BB, WØ,
OBN, PØ, QØ, AK, DIR, QW, QSN, QH1, QH2, ERRN,
ERH1, ERH2, ER12., SINE, COSN

Input:

ID — indicator
the values are to be stored in the disk if ID=1.
the values are to be read from the disk if ID = 2 or 3. If ID=3, the results are printed.

AXX, AD, AA, BB, WØ, OBN, PØ, QØ, AK, DIR, QW, QSN, QH1, QH2, ERRN, ERH1, ERH2, ER12, SINE, COSN, given in the main program.

1XØ — integer, the last integer used to create a random integer by RANDU.

IYØ — integer, the last integer used to create another random integer by RANDU.

KK time k at which all the result should be updated.

MM integer, (MM-1) values are stored for each QSN, QH1, QH2, ERRN, ERH1, ERH2, ER12 for the output purpose.

Output:

AXX, AD, AA, BB, WØ, PØ, QØ, AK, DIR, QW, QSN, QH1, QH2, ERRN, ERH1, ERH2, ER12, SINE, COSN, IXØ, IYØ, KK, MM.

Special Considerations

1. For the use of disk, variables A, B are used to store results with

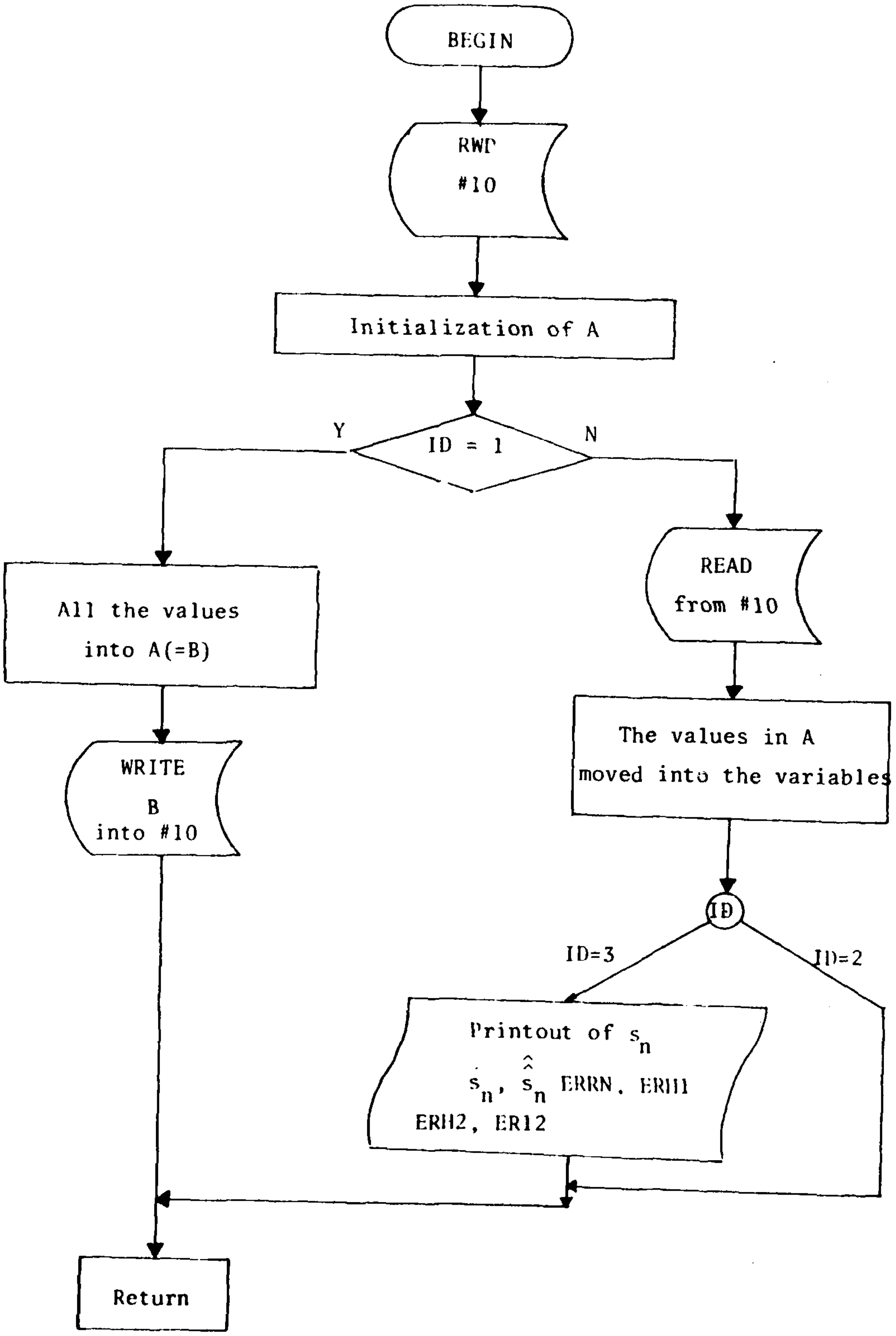
   A(1742), B(871, 2), EQUIVALENCE(A(1), B(1,1)).

2. All the values are stored in A in the following order.

A(1):	KK		A(168):	PØ(1,1,1)	(50)
A(2):	MM		A(218):	Q1(1,1,1)	(50)
A(3):	IXØ		A(268):	ERRN(1)	(50)
A(4):	IYØ		A(318):	ERH1(1)	(50)
A(5):	AK		A(368):	ERH2(1)	(50)
A(11):	AXX91,1,1)	(18)*	A(418):	ER12(1)	(50)
A(29):	AD(1,1)	(6)	A(468):	Blank	(100)
A(35):	AA(1,1,1)	(54)	A(568):	QSN(1,1)	(200)
A(89):	BB(1,1,1)	(54)	A(768):	QH1(1,1)	(200)
A(143):	RR(1)	(6)	A(968):	QH2(1,1)	(200)
A(149)	DIR(1)	(4)	A(1168):	Blank	(400)
A(153):	WØ(1)	(3)	A(1568):	SINE(1)	(33)
A(156):	QW(1)	(4)	A(1601):	COSN(1)	(33)
A(160):	QS(1)	(4)			
A(164):	QHØ(1)	(4)			

* 18 = 2x3x3 representing the array of AXX and A(11) corresponds to AX (1,1,1)

Flow Chart


BEGIN
RWD
#10
Initialization of A
Y
ID = 1
N
All the values
into A(=B)
READ
from #10
WRITE
B
into #10
The values in A
moved into the variables
ID
ID=3
ID=2
Printout of s_n
$\dot{s}_n$, $\hat{\hat{s}}_n$ ERRN, ERH1
ERH2, ER12
Return

TRIDMX

A subprogram

Purpose:

To transform a real symmetric matrix to tridiagonal form using Householder's method.

Calling sequence:

```
CALL     TRIDMX(N, NM, A, D, B)
```

Common Variables:

None,

Input:

N — number of rows and columns in matrix A Also N is the number of elements in D and B

NM — Maximum number of rows A can have as specified by the DIMENSION statement in the calling program.

A — NXN-dimensional array containing the symmetric matrix.

Output:

D — array containing the diagonal elements of the tridiagonal matrix.

B — array containing the off-diagonal elements of the tridiagonal elements of the tridiagonal matrix in locations B(2) through B(N). B(1)=0.0.

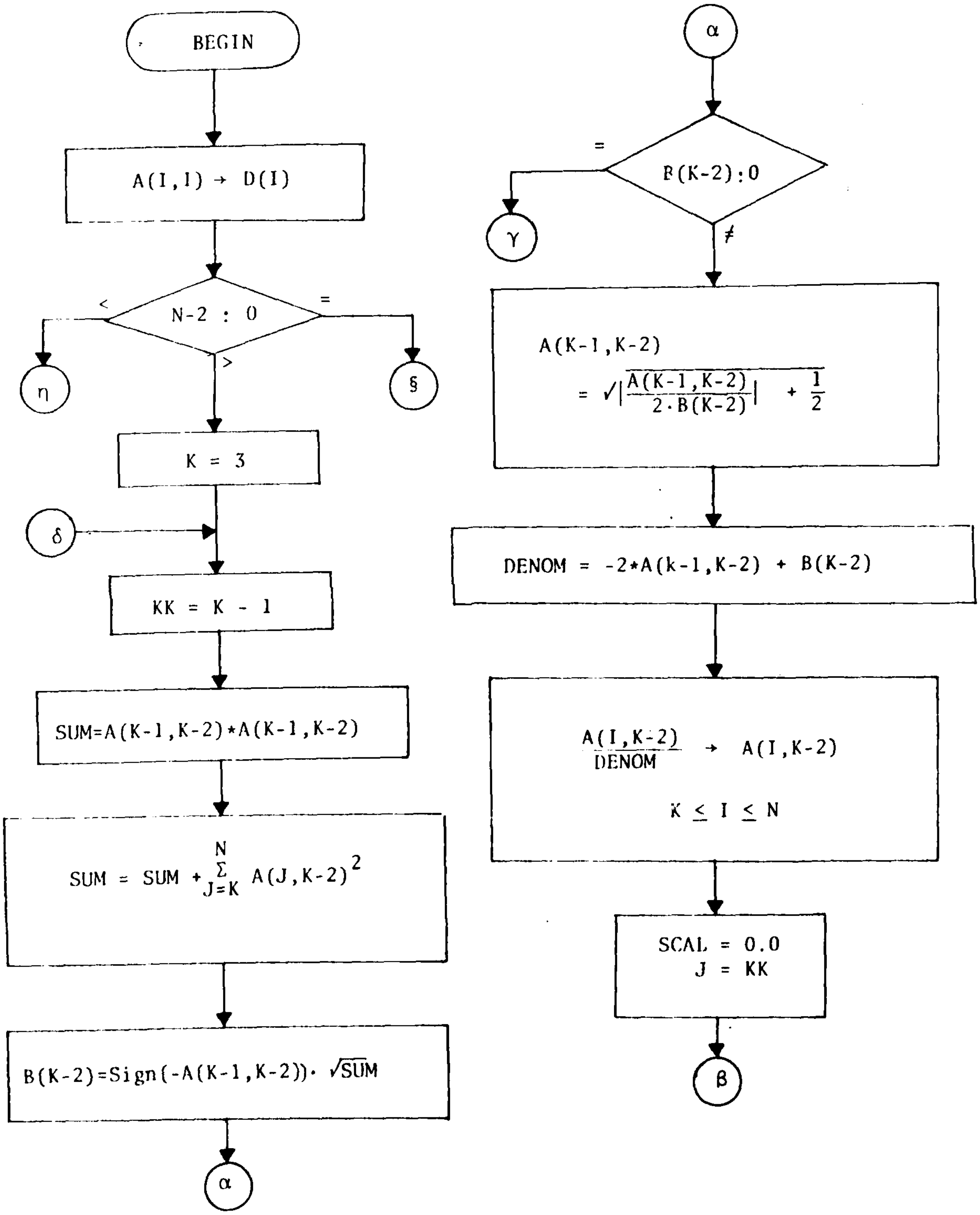
Special Considerations:

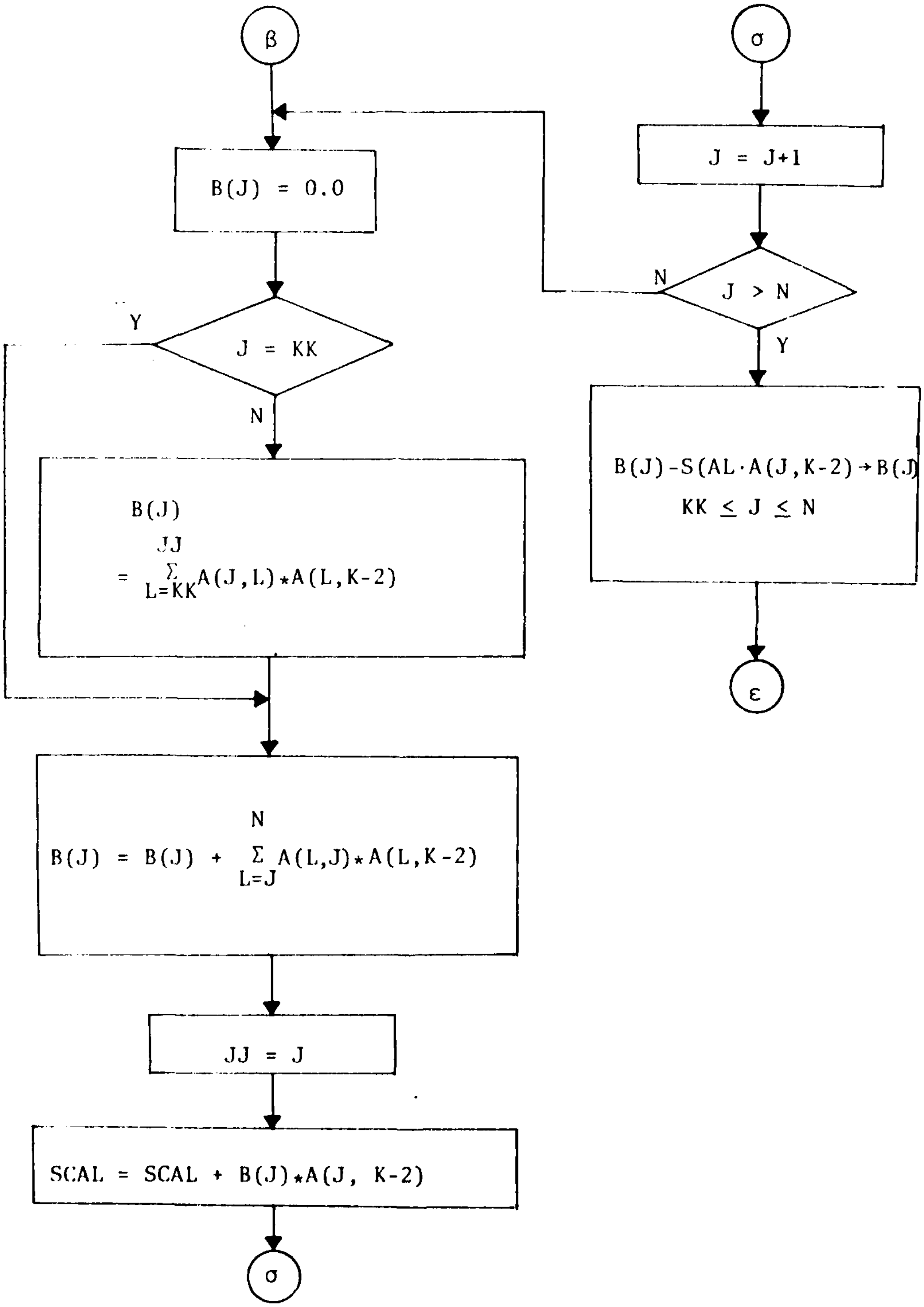
1. The lower triangular half of A is changed by TRIDMX.
2. TRIDMX is designed to be used with EIGVAL and EIGVEC.

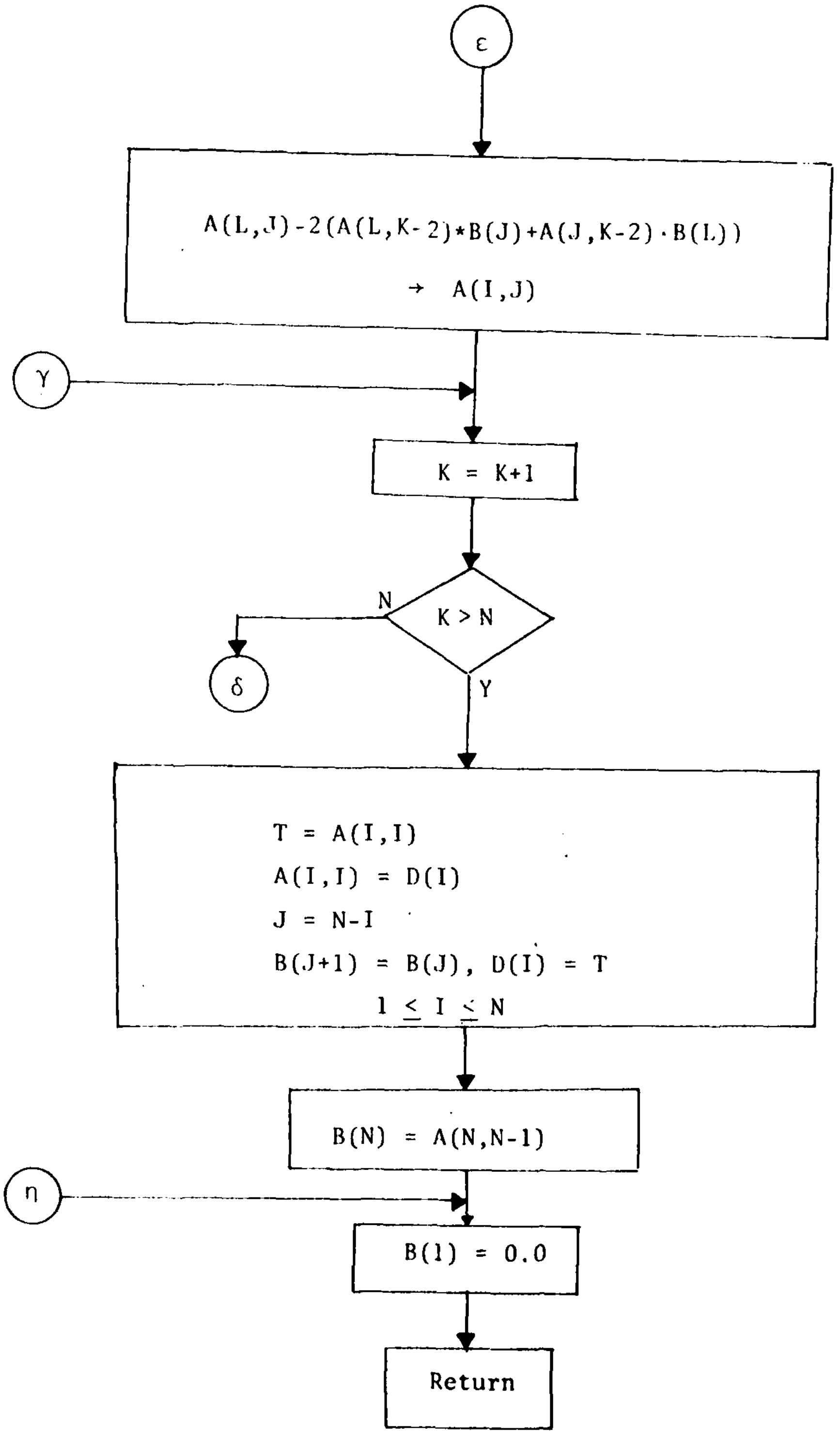
Other Subprograms Called.

None.

Flow Chart


BEGIN
A(I,I) → D(I)
N-2 : 0
<
=
>
η
§
K = 3
δ
KK = K - 1
SUM=A(K-1,K-2)*A(K-1,K-2)
$SUM = SUM + \sum_{J=K}^{N} A(J,K-2)^2$
B(K-2)=Sign(-A(K-1,K-2))· $\sqrt{SUM}$
α
α
B(K-2):0
=
γ
≠
A(K-1,K-2)
$= \sqrt{\left|\frac{A(K-1,K-2)}{2\cdot B(K-2)}\right| + \frac{1}{2}}$
DENOM = -2*A(k-1,K-2) + B(K-2)
$\frac{A(I,K-2)}{DENOM}$ → A(I,K-2)
K ≤ I ≤ N
SCAL = 0.0
J = KK
β


β
B(J) = 0.0
J = KK
Y
N
B(J) = $\sum_{L=KK}^{JJ}$ A(J,L)*A(L,K-2)
B(J) = B(J) + $\sum_{L=J}^{N}$ A(L,J)*A(L,K-2)
JJ = J
SCAL = SCAL + B(J)*A(J, K-2)
σ
σ
J = J+1
J > N
N
Y
B(J)-S(AL·A(J,K-2)→B(J)
KK ≤ J ≤ N
ε


ε
A(L,J)-2(A(L,K-2)*B(J)+A(J,K-2)·B(L))
→ A(I,J)
γ
K = K+1
K > N
N
δ
Y
T = A(I,I)
A(I,I) = D(I)
J = N-I
B(J+1) = B(J), D(I) = T
1 ≤ I ≤ N
B(N) = A(N,N-1)
η
B(1) = 0.0
Return

UPDWN

A subprogram:

Purpose:

To obtain maximum or minimum of two values.

Calling Sequence:

```
UPDWN(ID,X,Y)
```

Common Variables:

None.

Input:

ID	integer, if, ID=1, minimum, if, ID=2, maximum of two values X and Y to be computed.
X, Y	two values

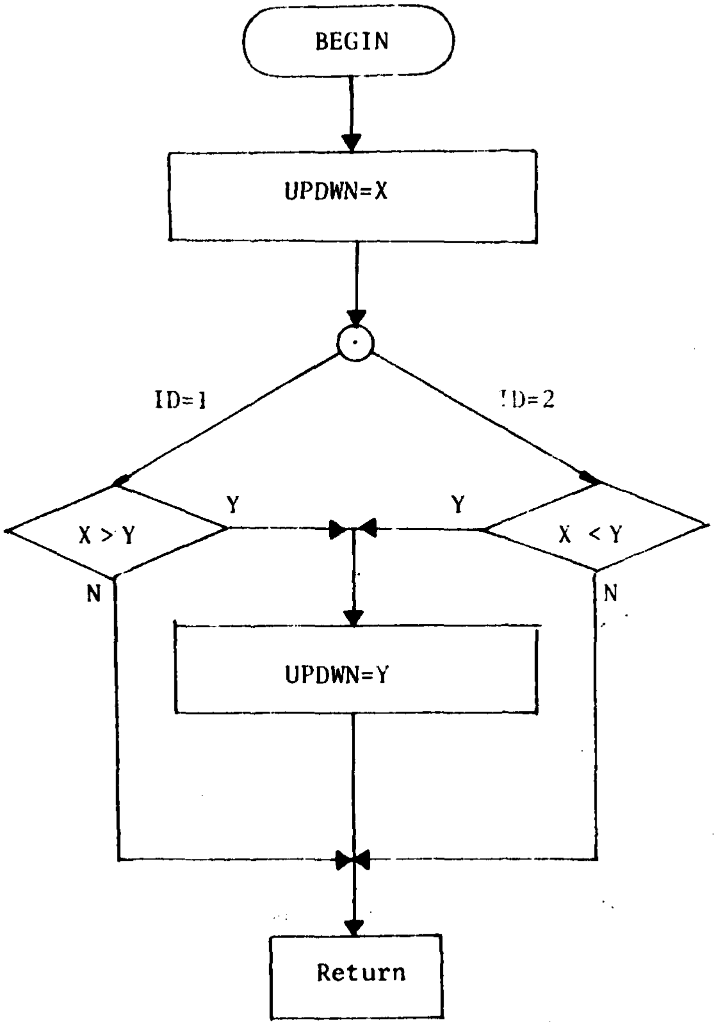
Special Consideration:

None.

Other Subprograms Called:

None.

Flow Chart



SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 79-0060 — 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) OPTIMAL ESTIMATION FOR THE SATELLITE ATTITUDE USING STAR TRACKER MEASUREMENTS	5. TYPE OF REPORT & PERIOD COVERED Interim 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) James Ting-Ho Lo	8. CONTRACT OR GRANT NUMBER(s) AFOSR 74-2671
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Maryland, Baltimore County Department of Mathematics Baltimore, Maryland 21228	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2304/A1
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research/NM Bolling AFB, Washington, DC 20332	12. REPORT DATE November 1978 13. NUMBER OF PAGES 167
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

optimal estimation, satellite attitude, star tracker, rate gyro, exponential Fourier density, HEAO, Kalman filter, simulation, recursive scheme, observability.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An optimal estimation scheme is presented, which determines the satellite attitude using the gyro readings and the star tracker measurements of a commonly used satellite attitude measuring unit. The scheme is mainly based on the exponential Fourier densities that have the desirable closure property under conditioning. By updating a finite and fixed number of parameters, the conditional probability density, which is an exponential Fourier density, is recursively determined. → next page

20. Abstract continued.

Simulation results indicate that the scheme is effective and robust. It is believed that this approach is applicable to many other attitude measuring units. As no linearization and approximation are necessary in the approach, it is ideal for systems involving high levels of randomness.

When a system involves little randomness and linearization is not expected to incur much error, the approach can provide a benchmark against which such suboptimal estimators as the extended Kalman filter and the least-squares estimator can be compared. In this spirit, simulated data for HEAO-A were processed to compare the optimal scheme and the extended Kalman filter. The results are presented.